ACTIONSCRIPT® 3.0 구성 요소 사용 설명서

마지막 업데이트 2011 년 5 월 13 일

법적 고지 사항

법적 고지 사항은 http://help.adobe.com/ko_KR/legalnotices/index.html을 참조하십시오.

목차

1장: 소개

Adobe® Flash® CS5 Professional은 효과가 뛰어난 웹 환경을 만들기 위한 표준 제작 도구입니다. 구성 요소는 이러한 환경을 제공하는 풍부한 인터넷 응용 프로그램을 구성하는 단위입니다. **구성 요소**는 Flash에서 제작하는 동안 또는 런타임에 Adobe® ActionScript® 메서드, 속성 및 이벤트를 사용하여 구성 요소를 사용자 정의하는 데 사용할 수 있는 매개 변수가 포함된 무비 클립입니다. 구성 요소는 개발자가 코드를 다시 사용하거나 공유할 수 있게 하고 복잡한 기능을 캡슐화하여 디자이너가 ActionScript를 사용하지 않고도 해당 기능을 사용하거나 사용자 정의할 수 있도록 디자인되었습니다.

구성 요소를 사용하면 일관된 모양과 비헤이비어를 가진 강력한 응용 프로그램을 쉽고 빠르게 만들 수 있습니다. 이 설명서에서는 Adobe ActionScript 3.0 구성 요소를 사용하여 응용 프로그램을 만드는 방법에 대해 설명합니다. **Adobe® ActionScript® 3.0 언어 및 구성 요소 참조 설명서**에서는 각 구성 요소의 API(Application Programming Interface)에 대해 설명합니다.

Adobe®에서 만든 구성 요소를 사용하거나 다른 개발자가 만든 구성 요소를 다운로드하거나 구성 요소를 직접 만들 수 있습니다.

이 설명서의 대상

이 설명서는 Flash 응용 프로그램을 만들 때 구성 요소를 사용하여 개발 시간을 단축하고자 하는 개발자를 위해 작성되었습니다. 즉, Flash에서 응용 프로그램을 개발하고 ActionScript를 작성하는 데 이미 능숙한 사용자를 대상으로 합니다.

ActionScript 작성에 익숙하지 않으면 문서에 구성 요소를 추가하고 속성 관리자나 [구성 요소 관리자]에서 매개 변수를 설정하고 [비헤이비어] 패널을 사용하여 이벤트를 처리할 수 있습니다. 예를 들어, ActionScript 코드를 작성하지 않고 버튼을 클릭할 때 웹 브라우저에서 URL을 여는 [웹 페이지로 이동] 비헤이비어를 Button 구성 요소에 연결할 수 있습니다.

더욱 강력한 응용 프로그램을 만들고자 하는 프로그래머는 동적으로 구성 요소를 만들고, ActionScript를 사용하여 런타임에 속성을 설정하고 메서드를 호출할 수 있습니다. 또한 이벤트 리스너 모델을 사용하여 이벤트를 처리할 수 있습니다.

자세한 내용은 15페이지의 “구성 요소를 사용한 작업”을 참조하십시오.

시스템 요구 사항

Flash 구성 요소는 Flash 시스템 요구 사항만 충족하면 사용할 수 있습니다.

Flash CS3 이상 구성 요소를 사용하는 모든 SWF 파일은 Adobe® Flash® Player 9.0.28.0 이상을 사용하여 봐야 하며, ActionScript 3.0용으로 제작해야 합니다([파일] > [제작 설정]의 [Flash] 탭에서 설정 가능).

설명서

이 설명서에서는 구성 요소를 사용하여 Flash 응용 프로그램을 개발하는 방법에 대해 자세하게 설명합니다. 이 설명서는 Flash와 ActionScript 3.0에 대한 일반적인 지식이 있는 독자를 대상으로 작성되었습니다. Flash 및 관련 제품에 대한 설명서는 별도로 제공됩니다.

이 문서는 PDF 파일이나 온라인 도움말 형태로 사용할 수 있습니다. 온라인 도움말을 보려면 Flash를 시작하고 [도움말] > [Flash 도움말] > [Adobe ActionScript 3.0 구성 요소 사용 설명서]를 선택하십시오.

Flash에 대한 자세한 내용은 다음 문서를 참조하십시오.

- **Flash 사용**
- **ActionScript 3.0 개발자 안내서**

- **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**

인쇄 규칙

이 설명서는 다음과 같은 인쇄 규칙을 따릅니다.

- 기울임체 글꼴은 대체해야 하는 값을 나타냅니다(예: 폴더 경로).
- 코드 글꼴은 메서드 및 속성 이름을 포함한 ActionScript 코드를 나타냅니다.
- 코드 기울임체 글꼴은 대체해야 하는 코드 항목을 나타냅니다(예: ActionScript 매개 변수).
- **굵은 글꼴**은 사용자가 입력하는 값을 나타냅니다.

설명서에 사용된 용어

이 설명서에는 다음과 같은 용어가 사용됩니다.

런타임 Flash Player에서 코드가 실행 중인 상태를 나타냅니다.

제작하는 동안 Flash 제작 환경에서 작업 중인 상태를 나타냅니다.

추가 리소스

Adobe에서는 이 설명서의 내용 이외에도 Adobe 개발자 센터 및 Adobe 디자인 센터를 통해 정기적으로 업데이트되는 문서, 디자인 아이디어 및 예제를 제공합니다.

추가 구성 요소 샘플은 www.adobe.com/go/learn_fl_samples_kr에서 다운로드할 수 있습니다.

Adobe 개발자 센터
Adobe 개발자 센터는 ActionScript에 대한 최신 정보, 실제 응용 프로그램 개발 관련 기사 및 중요하게 부각되고 있는 이슈에 대한 정보를 제공하는 리소스입니다. 개발자 센터는 www.adobe.com/go/flash_devcenter_kr에 있습니다.

Adobe 디자인 센터
최신 디지털 디자인 및 모션 그래픽을 만나보십시오. 주요 아티스트별로 작품을 찾아보고 새로운 디자인 흐름을 살펴보며 자습서, 핵심 작업 과정 및 고급 기술을 통해 기술 향상을 도모할 수 있습니다. 한 달에 두 번씩, 새로운 자습서와 기사 및 창조적 영감을 불러일으키는 갤러리 작품들이 전시됩니다. 디자인 센터는 www.adobe.com/go/fl_designcenter_kr에 있습니다.

2장: ActionScript 3.0 구성 요소

Adobe® Flash® Professional CS5 구성 요소는 매개 변수를 사용하여 모양과 비헤이비어를 수정할 수 있는 동영상 클립입니다. 구성 요소는 RadioButton 또는 CheckBox와 같은 간단한 사용자 인터페이스 컨트롤이거나, List 또는 DataGrid와 같이 내용을 포함할 수 있습니다.

구성 요소를 사용하면 일관된 모양과 비헤이비어를 가진 강력한 Flash 응용 프로그램을 쉽고 빠르게 만들 수 있습니다. 사용자 정의 버튼, 콤보 상자 및 목록을 직접 만드는 대신 이러한 컨트롤을 구현하는 Flash 구성 요소를 사용할 수 있습니다. [구성 요소] 패널에서 응용 프로그램 문서로 구성 요소를 드래그하면 됩니다. 또한 응용 프로그램 디자인에 맞도록 해당 구성 요소의 모양과 느낌을 손쉽게 사용자 정의할 수 있습니다.

ActionScript에 대한 고급 지식 없이도 이러한 작업을 모두 수행할 수 있을 뿐만 아니라 ActionScript 3.0을 사용하여 구성 요소의 비헤이비어를 수정하거나 새로운 비헤이비어를 구현할 수도 있습니다. 각 구성 요소에는 **API**(Application Programming Interface)를 구성하는 고유한 ActionScript 메서드, 속성 및 이벤트 집합이 있습니다. 이러한 API를 사용하면 응용 프로그램이 실행 중인 동안에 구성 요소를 만들고 조작할 수 있습니다.

API를 사용하면 사용자 정의된 새로운 구성 요소도 만들 수 있습니다. Adobe Exchange(www.adobe.com/go/flash_exchange_kr)에서는 Flash 커뮤니티 회원이 만든 구성 요소를 다운로드할 수 있습니다. 구성 요소 만들기에 대한 자세한 내용은 www.adobe.com/go/learn_fl_creating_components_kr를 참조하십시오.

ActionScript 3.0 구성 요소 아키텍처에는 모든 구성 요소의 기반이 되는 클래스, 모양을 사용자 정의하는 데 사용할 수 있는 스킨과 스타일, 이벤트 처리 모델, 포커스 관리, 액세스 가능성 인터페이스 등이 포함됩니다.

참고: Adobe Flash CS5에는 ActionScript 3.0 구성 요소뿐 아니라 ActionScript 2.0 구성 요소도 포함되어 있습니다. 이 두 가지 구성 요소 집합을 혼합해서 사용할 수는 없습니다. 특정 응용 프로그램에 대해 한 가지 구성 요소 집합만 사용해야 합니다. Flash CS5는 ActionScript 2.0 파일을 여는지, 아니면 ActionScript 3.0 파일을 여는지에 따라 ActionScript 2.0 구성 요소 또는 ActionScript 3.0 구성 요소를 제공합니다. 따라서 새로운 Flash 문서를 만들 때는 [Flash 파일(ActionScript 3.0)] 또는 [Flash 파일(ActionScript 2.0)] 중 하나를 지정해야 합니다. 기존 문서를 열 경우에는 [제작 설정]에 따라 사용할 구성 요소 집합이 결정됩니다. ActionScript 2.0 구성 요소에 대한 자세한 내용은 Adobe® ActionScript® 2.0 구성 요소 사용을 참조하십시오.

Flash ActionScript 3.0 구성 요소의 전체 목록은 4페이지의 “구성 요소 유형”을 참조하십시오.

구성 요소 사용의 장점

구성 요소를 사용하면 코딩 과정에서 응용 프로그램의 디자인 과정을 분리할 수 있습니다. 따라서 개발자가 응용 프로그램에 사용할 수 있는 디자이너용 기능을 만들 수 있습니다. 개발자는 자주 사용하는 기능을 구성 요소에 캡슐화할 수 있으며 디자이너는 매개 변수를 변경하여 구성 요소의 크기, 위치 및 비헤이비어를 사용자 정의할 수 있습니다. 디자이너는 또한 그래픽 요소나 스킨을 편집하여 구성 요소의 모양을 변경할 수도 있습니다.

구성 요소는 스타일, 스킨 및 포커스 관리와 같은 핵심 기능을 공유합니다. 응용 프로그램에 첫 번째 구성 요소를 추가하면 이 핵심 기능이 20KB 정도의 크기를 차지합니다. 그리고 다른 구성 요소를 추가하면 첫 번째 구성 요소의 핵심 기능에 사용된 메모리 공간이 이후 구성 요소에도 공유되므로 응용 프로그램 크기를 줄일 수 있습니다.

이 단원에서는 ActionScript 3.0 구성 요소의 몇 가지 장점에 대해 간략하게 설명합니다.

강력해진 ActionScript 3.0 Flash Player 기능 향상에 중요한 단계인 강력한 객체 지향 프로그래밍 언어를 제공합니다. 이 언어는 재사용이 가능한 코드 베이스에서 풍부한 인터넷 응용 프로그램을 만들 수 있도록 설계되었습니다. ActionScript 3.0은 스크립팅에 대한 국제 표준 언어인 ECMAScript를 기반으로 하며 ECMAScript(ECMA-262) 버전 3 언어 사양을 따릅니다. ActionScript 3.0에 대한 자세한 소개는 **ActionScript 3.0 개발자 안내서**를 참조하십시오. 언어에 대한 참조 정보는 ActionScript 3.0 Reference를 참조하십시오.

FLA 기반 사용자 인터페이스 구성 요소 제작하는 동안 스킨에 손쉽게 액세스하여 이를 사용자 정의할 수 있게 해 줍니다. 이 구성 요소는 또한 구성 요소 모양을 사용자 정의하고 런타임에 스킨을 로드하는 데 사용할 수 있는 스킨 스타일 등의 스타일도 제공합니다. 자세한 내용은 89페이지의 "UI 구성 요소 사용자 정의" 및 ActionScript 3.0 Reference를 참조하십시오.

새로운 FVLPlayback 구성 요소를 통한 FLVPlaybackCaptioning 구성 요소 추가 이와 더불어 전체 화면 지원, 향상된 실시간 미리 보기, 색상 및 알파 설정을 추가하는 데 사용할 수 있는 스킨 및 향상된 FLV 다운로드 및 레이아웃 기능도 제공합니다.

속성 관리자 및 구성 요소 관리자 Flash에서 제작하는 동안 구성 요소 매개 변수를 변경할 수 있게 해 줍니다. 자세한 내용은 17페이지의 "구성 요소 파일을 사용한 작업" 및 19페이지의 "매개 변수 및 속성 설정"을 참조하십시오.

새 컬렉션 대화 상자 ComboBox, List 및 TileList 구성 요소의 새 컬렉션 대화 상자에서는 사용자 인터페이스를 통해 dataProvider 속성을 채울 수 있습니다. 자세한 내용은 26페이지의 "DataProvider 만들기"를 참조하십시오.

ActionScript 3.0 이벤트 모델 응용 프로그램에서 이벤트를 수신하고 이벤트 핸들러를 호출하여 이벤트에 응답할 수 있게 해 줍니다. 자세한 내용은 8페이지의 "ActionScript 3.0 이벤트 처리 모델" 및 21페이지의 "이벤트 처리"를 참조하십시오.

관리자 클래스 응용 프로그램에서 손쉬운 방법으로 포커스를 처리하고 스타일을 관리할 수 있게 해 줍니다. 자세한 내용은 ActionScript 3.0 Reference를 참조하십시오.

UIComponent 기본 클래스 이 클래스를 확장하는 구성 요소에 기본 메서드, 속성 및 이벤트를 제공합니다. ActionScript 3.0 사용자 인터페이스 구성 요소는 모두 UIComponent 클래스에서 상속됩니다. 자세한 내용은 ActionScript 3.0 Reference에서 UIComponent 클래스를 참조하십시오.

SWC 사용 UI FLA 기반 구성 요소의 SWC를 사용하면 ActionScript 정의가 구성 요소 타임라인 내의 에셋으로 제공되므로 컴파일 속도가 빨라집니다.

확장이 용이한 클래스 계층 구조 ActionScript 3.0을 사용하는 이 클래스 계층 구조를 통해 고유한 네임스페이스를 만들고 필요에 따라 클래스를 가져오고 하위 클래스를 만들어 쉽게 구성 요소를 확장할 수 있습니다.

자세한 내용은 ActionScript 3.0 Reference를 참조하십시오.

참고: Flash CS5는 FLA 기반 구성 요소와 SWC 기반 구성 요소를 모두 지원합니다. 자세한 내용은 15페이지의 "구성 요소 아키텍처"을 참조하십시오.

구성 요소 유형

Flash 구성 요소는 Flash CS5를 설치할 때 함께 설치합니다.

ActionScript 3.0 구성 요소에 포함된 UI(사용자 인터페이스) 구성 요소는 다음과 같습니다.

Button	List	TextArea
CheckBox	NumericStepper	TextInput
ColorPicker	RadioButton	TileList
ComboBox	ProgressBar	UILoader
DataGrid	ScrollPane	UIScrollBar
Label	Slider	

Flash ActionScript 3.0 구성 요소에는 사용자 인터페이스 구성 요소 외에도 다음과 같은 구성 요소와 지원 클래스가 포함됩니다.

- FLVPlayback 구성 요소(fl.video.FLVPlayback) - SWC 기반 구성 요소입니다.

  FLVPlayback 구성 요소를 사용하면 Flash 응용 프로그램에 비디오 플레이어를 쉽게 추가할 수 있습니다. 이 비디오 플레이어는 Adobe® FVSS(Flash® Video Streaming Service) 또는 Adobe Macromedia® FMS(Flash® Media Server)에서 HTTP를 통해 점진적 스트리밍 비디오를 재생할 수 있습니다. 자세한 내용은 122페이지의 “FLVPlayback 구성 요소 사용”을 참조하십시오.

- FLVPlayback 사용자 정의 UI 구성 요소 - FLA 기반 구성 요소로서 FLVPlayback 구성 요소의 ActionScript 2.0 및 ActionScript 3.0 버전에서 사용할 수 있습니다. 자세한 내용은 122페이지의 “FLVPlayback 구성 요소 사용”을 참조하십시오.

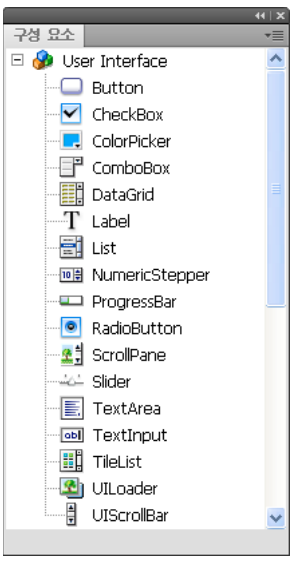
- FLVPlayback Captioning 구성 요소 - FLVPlayback에 대한 클로즈드 캡션을 제공합니다. 155페이지의 “FLVPlayback Captioning 구성 요소 사용”을 참조하십시오.

  ActionScript 3.0 구성 요소 및 구성 요소별 지원 클래스의 전체 목록은 ActionScript 3.0 Reference를 참조하십시오.

Flash 구성 요소 보기:

[구성 요소] 패널에서 Flash ActionScript 3.0 구성 요소를 보려면 다음 단계를 수행하십시오.

1. Flash를 시작합니다.
2. 새 Flash 파일(ActionScript 3.0)을 만들거나 [제작 설정]이 ActionScript 3.0으로 지정된 기존 Flash 문서를 엽니다.
3. [구성 요소] 패널이 열려 있지 않은 경우 [윈도우] > [구성 요소]를 선택하여 [구성 요소] 패널을 엽니다.



사용자 인터페이스 구성 요소가 포함된 구성 요소 패널

추가 구성 요소는 Adobe Exchange(www.adobe.com/go/flash_exchange_kr)에서 다운로드할 수도 있습니다. Exchange에서 다운로드한 구성 요소를 설치하려면 www.adobe.com/go/exchange_kr에서 Adobe® Extension Manager를 다운로드하여 설치하십시오. Adobe Exchange 홈 링크를 클릭하여 Extension Manager 링크를 찾아보십시오.

모든 구성 요소는 Flash의 [구성 요소] 패널에 나타날 수 있습니다. Windows® 또는 Macintosh® 컴퓨터에서 구성 요소를 설치하려면 다음 단계를 수행하십시오.

Windows 기반 컴퓨터나 Macintosh 컴퓨터에 구성 요소 설치

1. Flash를 종료합니다.

2 구성 요소가 들어 있는 SWC 또는 FLA 파일을 하드 디스크의 다음 폴더로 이동합니다.

- Windows:

  C:\Program Files\Adobe\Adobe Flash CS5**language**\Configuration\Components

- Macintosh:

  Macintosh HD:Applications:Adobe Flash CS5:Configuration:Components

3 Flash를 시작합니다.

4 [구성 요소] 패널이 열려 있지 않은 경우 [윈도우] > [구성 요소]를 선택하여 [구성 요소] 패널에서 구성 요소를 봅니다.

구성 요소 파일에 대한 자세한 내용은 17페이지의 "구성 요소 파일을 사용한 작업"을 참조하십시오.

문서에서 구성 요소 추가 및 삭제

FLA 기반 구성 요소를 [구성 요소] 패널에서 스테이지로 드래그하면 Flash에서는 편집 가능한 동영상 클립을 라이브러리로 가져옵니다. SWC 기반 구성 요소를 스테이지로 드래그하면 Flash에서는 컴파일된 클립을 라이브러리로 가져옵니다. 구성 요소를 라이브러리로 가져온 후에는 구성 요소 인스턴스를 [라이브러리] 패널이나 [구성 요소] 패널에서 스테이지로 드래그할 수 있습니다.

제작하는 동안 구성 요소 추가

구성 요소를 문서에 추가하려면 [구성 요소] 패널에서 드래그하십시오. 그런 다음 속성 관리자나 [구성 요소 관리자]의 [매개 변수] 탭에서 구성 요소 인스턴스 각각에 대한 속성을 설정할 수 있습니다.

1 [윈도우] > [구성 요소]를 선택합니다.

2 [구성 요소] 패널에서 구성 요소를 두 번 클릭하거나 구성 요소를 스테이지로 드래그합니다.

3 스테이지에서 구성 요소를 선택합니다.

4 속성 관리자가 표시되어 있지 않은 경우에는 [윈도우] > [속성] > [속성]을 선택합니다.

5 속성 관리자에서 구성 요소 인스턴스의 이름을 입력합니다.

6 [윈도우] > [구성 요소 관리자]를 선택하고 [매개 변수] 탭을 선택하여 인스턴스의 매개 변수를 지정합니다.

자세한 내용은 19페이지의 "매개 변수 및 속성 설정"을 참조하십시오.

7 폭(W:) 및 높이(H:) 값을 편집하여 구성 요소의 크기를 원하는 대로 변경합니다.

특정 구성 요소 유형의 크기 조절에 대한 자세한 내용은 89페이지의 "UI 구성 요소 사용자 정의"을 참조하십시오.

8 [컨트롤] > [동영상 테스트]를 선택하거나 Control+Enter를 눌러 문서를 컴파일하고 설정 결과를 확인합니다.

구성 요소에 대한 스타일 속성을 설정하여 구성 요소의 색상과 텍스트 서식을 변경하거나 구성 요소의 스킨을 편집하여 모양을 사용자 정의할 수도 있습니다. 이 항목에 대한 자세한 내용은 89페이지의 "UI 구성 요소 사용자 정의"를 참조하십시오.

제작하는 동안 구성 요소를 스테이지로 드래그한 경우 해당 인스턴스 이름(예: myButton)을 사용하여 구성 요소를 참조할 수 있습니다.

ActionScript를 사용하여 런타임에 구성 요소 추가

ActionScript를 사용하여 런타임에 구성 요소를 문서에 추가하려면 SWF 파일을 컴파일할 때 응용 프로그램의 라이브러리([윈도우] > [라이브러리])에 해당 구성 요소가 있어야 합니다. 라이브러리에 구성 요소를 추가하려면 [구성 요소] 패널의 구성 요소를 [라이브러리] 패널로 드래그합니다. 라이브러리에 대한 자세한 내용은 20페이지의 "라이브러리"를 참조하십시오.

응용 프로그램에서 구성 요소의 API를 사용하려면 해당 구성 요소의 클래스 파일도 가져와야 합니다. 구성 요소 클래스 파일은 하나 이상의 클래스가 포함된 패키지에 설치됩니다. 구성 요소 클래스를 가져오려면 import 문을 사용하여 패키지 이름과 클래스 이름을 지정하십시오. 예를 들어, 다음 import 문을 사용하면 Button 클래스를 가져올 수 있습니다.

```
import fl.controls.Button;
```

구성 요소를 포함하는 패키지에 대한 자세한 내용은 ActionScript 3.0 Reference를 참조하십시오. 구성 요소 소스 파일의 위치에 대한 자세한 내용은 17페이지의 "구성 요소 파일을 사용한 작업"을 참조하십시오.

구성 요소 인스턴스를 만들려면 구성 요소의 ActionScript 생성자 메서드를 호출해야 합니다. 예를 들어, 다음 명령문은 aButton이라는 Button 인스턴스를 만듭니다.

```
var aButton:Button = new Button();
```

마지막으로 정적 addChild() 메서드를 호출하여 구성 요소 인스턴스를 스테이지 또는 응용 프로그램 컨테이너에 추가합니다. 예를 들어, 다음 명령문은 aButton 인스턴스를 추가합니다.

```
addChild(aButton);
```

이때 구성 요소 API를 사용하여 스테이지에 있는 구성 요소의 크기와 위치를 동적으로 지정하고, 이벤트를 수신하고, 구성 요소의 비헤이비어를 수정하도록 속성을 설정할 수 있습니다. 특정 구성 요소의 API에 대한 자세한 내용은 ActionScript 3.0 Reference를 참조하십시오.

addChild() 메서드에 대한 자세한 내용은 22페이지의 "표시 목록을 사용한 작업"을 참조하십시오.

구성 요소 삭제

제작하는 동안 스테이지에서 구성 요소 인스턴스를 삭제하려면 구성 요소를 선택한 다음 Delete 키를 누르면 됩니다. 이 경우 스테이지에서 인스턴스가 제거될 뿐이며 응용 프로그램에서 구성 요소가 제거되지는 않습니다.

구성 요소를 스테이지나 라이브러리에 배치한 후 Flash 문서에서 구성 요소를 삭제하려면 라이브러리에서 구성 요소 및 관련 에셋을 삭제해야 하며, 스테이지에서 구성 요소를 삭제하는 것만으로는 부족합니다. 라이브러리에서 구성 요소를 제거하지 않으면 컴파일 시 해당 구성 요소가 응용 프로그램에 포함됩니다.

1. [라이브러리] 패널에서 구성 요소의 심볼을 선택합니다.
2. [라이브러리] 패널의 아래쪽에 있는 [삭제] 버튼을 클릭하거나 [라이브러리] 패널 메뉴에서 [삭제]를 선택합니다.

   이러한 단계를 반복하여 구성 요소와 관련 에셋을 삭제합니다.

   응용 프로그램이 실행 중일 때 구성 요소를 컨테이너에서 제거하는 방법에 대한 자세한 내용은 24페이지의 "표시 목록에서 구성 요소 제거"을 참조하십시오.

구성 요소 버전 찾기

Flash ActionScript 3.0 구성 요소에는 Adobe 기술 지원 센터에 해당 정보를 제공해야 하는 경우나 사용 중인 구성 요소의 버전을 확인해야 하는 경우에 표시할 수 있는 버전 속성이 있습니다.

사용자 인터페이스 구성 요소의 버전 번호 표시

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. 구성 요소를 스테이지로 드래그한 다음 인스턴스 이름을 지정합니다. 예를 들어, ComboBox를 스테이지로 드래그하고 **aCb**라는 이름을 지정합니다.
3. **F9** 키를 누르거나 [윈도우] > [액션]을 선택하여 [액션] 패널을 엽니다.
4. 기본 타임라인에서 [프레임 1]을 클릭하고 [액션] 패널에 다음 코드를 추가합니다.

   ```
   trace(aCb.version);
   ```

다음 그림의 버전 번호와 비슷한 버전 번호가 [출력] 패널에 표시되어야 합니다.

FLVPlayback 및 FLVPlaybackCaptioning 구성 요소의 경우 버전 번호가 클래스 상수에 저장되므로 인스턴스 이름 대신 클래스 이름을 참조해야 합니다.

FLVPlayback 및 FLVPlaybackCaptioning 구성 요소의 버전 번호 표시

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. FLVPlayback 구성 요소와 FLVPlaybackCaptioning 구성 요소를 [라이브러리] 패널로 드래그합니다.
3. **F9** 키를 누르거나 [윈도우] > [액션]을 선택하여 [액션] 패널을 엽니다.
4. 기본 타임라인에서 [프레임 1]을 클릭하고 [액션] 패널에 다음 코드를 추가합니다.

```
import fl.video.*;
trace("FLVPlayback.VERSION: " + FLVPlayback.VERSION);
trace("FLVPLaybackCaptioning.VERSION: " + FLVPlaybackCaptioning.VERSION);
```

[출력] 패널에 버전 번호가 나타납니다.

ActionScript 3.0 이벤트 처리 모델

ActionScript 3.0에서는 이전 버전의 ActionScript에 사용된 다양한 이벤트 처리 메커니즘을 대체하는 단일 이벤트 처리 모델을 사용합니다. 새 이벤트 모델은 DOM(Document Object Model) 레벨 3 이벤트 사양에 기반합니다.

ActionScript 2.0 addListener() 메서드를 사용해 본 경험이 있는 개발자의 경우 ActionScript 2.0 이벤트 리스너 모델과 ActionScript 3.0 이벤트 모델 간의 차이점을 살펴보면 도움이 될 것입니다. 다음 목록에서는 두 이벤트 모델의 몇 가지 주요 차이점을 설명합니다.

- ActionScript 2.0에서는 이벤트 리스너를 추가하는 경우 상황에 따라 addListener() 또는 addEventListener()를 사용하지만 ActionScript 3.0에서는 항상 addEventListener()를 사용합니다.
- ActionScript 2.0에는 이벤트 흐름이 없으므로 이벤트를 브로드캐스트하는 객체에서만 addListener() 메서드를 호출할 수 있지만 ActionScript 3.0에서는 이벤트 흐름에 포함된 모든 객체에서 addEventListener() 메서드를 호출할 수 있습니다.
- ActionScript 2.0에서는 함수, 메서드 또는 객체를 이벤트 리스너로 사용할 수 있지만 ActionScript 3.0에서는 함수 또는 메서드만 이벤트 리스너로 사용할 수 있습니다.
- on(**event**) 구문은 이제 ActionScript 3.0에서 지원되지 않으므로 ActionScript 이벤트 코드를 동영상 클립에 첨부할 수 없습니다. 이벤트 리스너를 추가할 때는 addEventListener()만 사용할 수 있습니다.

  다음은 aButton이라는 Button 구성 요소에서 MouseEvent.CLICK 이벤트를 수신하는 예제로서 기본적인 ActionScript 3.0 이벤트 처리 모델을 보여 줍니다.

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler);
function clickHandler(event:MouseEvent):void {
trace("clickHandler detected an event of type: " + event.type);
trace("the event occurred on: " + event.target.name);
}
```

  ActionScript 3.0 이벤트 처리에 대한 자세한 내용은 **ActionScript 3.0** 프로그래밍을 참조하십시오. 구성 요소의 ActionScript 3.0 이벤트 처리에 대한 자세한 내용은 21페이지의 "이벤트 처리"을 참조하십시오.

간단한 응용 프로그램

이 단원에서는 Flash 구성 요소와 Flash 제작 도구를 사용하여 간단한 ActionScript 3.0 응용 프로그램을 만드는 방법을 단계별로 설명합니다. 여기서 사용하는 예제는 타임라인에 ActionScript 코드가 포함된 FLA 파일로 제공됩니다. 또한 이 예제는 FLA 파일의 라이브러리에 구성 요소만 포함된 외부 ActionScript 클래스 파일로도 제공됩니다. 일반적으로 크기가 큰 응용 프로그램의 경우 외부 클래스 파일을 사용하여 개발하면 클래스와 응용 프로그램 간에 코드를 공유하고 응용 프로그램을 보다 쉽게 유지 관리할 수 있습니다. ActionScript 3.0을 사용한 프로그래밍에 대한 자세한 내용은 **ActionScript 3.0 프로그래밍**을 참조하십시오.

응용 프로그램 디자인

ActionScript 구성 요소 응용 프로그램에 대한 첫 번째 예제는 표준 "Hello World" 응용 프로그램을 변형한 것으로 이에 대한 설계는 매우 간단합니다.

- 이 응용 프로그램을 Greetings라고 하겠습니다.
- 이 응용 프로그램은 TextArea를 사용하여 먼저 Hello World라는 인사말을 표시합니다.
- 이 응용 프로그램은 ColorPicker를 사용하여 텍스트 색상을 변경합니다.
- 이 응용 프로그램은 세 가지 RadioButton을 사용하여 텍스트 크기를 작게, 크게 또는 가장 크게 설정합니다.
- 이 응용 프로그램은 ComboBox를 사용하여 드롭 다운 목록에서 다른 인사말을 선택합니다.
- 이 응용 프로그램은 [구성 요소] 패널의 구성 요소를 사용하며 ActionScript 코드를 통해 응용 프로그램 요소도 만듭니다.

이러한 정의를 토대로 응용 프로그램 작성을 시작할 수 있습니다.

Greetings 응용 프로그램 만들기

다음은 Flash 제작 도구로 FLA 파일을 만들고, 스테이지에 구성 요소를 배치하고, 타임라인에 ActionScript 코드를 추가하여 Greetings 응용 프로그램을 만드는 단계입니다.

FLA 파일에서 Greetings 응용 프로그램 만들기:

1. [파일] > [새로 만들기]를 선택합니다.
2. [새 문서] 대화 상자에서 [Flash 파일(ActionScript 3.0)]을 선택하고 [확인]을 클릭합니다.

   새 Flash 윈도우가 열립니다.
3. [파일] > [저장]을 선택하고 Flash 파일의 이름을 **Greetings.fla**로 지정한 다음 [저장] 버튼을 클릭합니다.
4. Flash [구성 요소] 패널에서 TextArea 구성 요소를 선택하고 스테이지로 드래그합니다.
5. [속성] 윈도우에서 스테이지에 TextArea 영역이 선택되어 있는 상태로 **aTa**를 인스턴스 이름으로 입력하고 다음 정보를 입력합니다.
   - W 값(폭)으로 **230**을 입력합니다.
   - H 값(높이)으로 **44**를 입력합니다.
   - X 값(가로 위치)으로 **165**를 입력합니다.
   - Y 값(세로 위치)으로 **57**을 입력합니다.
   - [매개 변수] 탭에서 텍스트 매개 변수로 **Hello World!**를 입력합니다.
6. ColorPicker 구성 요소를 스테이지로 드래그하여 TextArea 왼쪽에 배치한 후 인스턴스 이름으로 **txtCp** 를 지정합니다. 속성 관리자에서 다음 정보를 입력합니다.
   - X 값으로 **96**을 입력합니다.
   - Y 값으로 **72**를 입력합니다.

7 세 가지 RadioButton 구성 요소를 한 번에 하나씩 스테이지로 드래그하고 인스턴스 이름을 각각 **smallRb**, **largerRb** 및 **largestRb**로 지정합니다. 속성 관리자에서 다음 정보를 입력합니다.

- 각각의 W 값으로 **100**을, H 값으로 **22**를 입력합니다.
- X 값으로 **155**를 입력합니다.
- Y 값으로 smallRb는 **120**, largerRb는 **148**, largestRb는 **175**를 각각 입력합니다.
- 각각에 대해 groupName 매개 변수로 **fontRbGrp**를 입력합니다.
- [매개 변수] 탭에서 각각의 레이블을 **Small**, **Larger**, **Largest**로 입력합니다.

8 ComboBox를 스테이지로 드래그하고 인스턴스 이름을 **msgCb**로 지정합니다. 속성 관리자에서 다음 정보를 입력합니다.

- W 값으로 **130**을 입력합니다.
- X 값으로 **265**를 입력합니다.
- Y 값으로 **120**을 입력합니다.
- [매개 변수] 탭에서 prompt 매개 변수로 **Greetings**를 입력합니다.
- dataProvider 매개 변수의 텍스트 필드를 두 번 클릭하여 [값] 대화 상자를 엽니다.
- 더하기 기호를 클릭하고 레이블 값을 **Hello World!**로 바꿉니다.
- 위의 단계를 반복하여 레이블 값 **Have a nice day!**와 **Top of the Morning!**을 추가합니다.
- [확인]을 클릭하여 [값] 대화 상자를 닫습니다.

9 파일을 저장합니다.

10 **F9** 키를 누르거나 [윈도우] 메뉴에서 [액션]을 선택하여 [액션] 패널을 엽니다(열려 있지 않은 경우). 기본 타임라인에서 [프레임 1]을 클릭하고 [액션] 패널에서 다음 코드를 입력합니다.

```
import flash.events.Event;
import fl.events.ComponentEvent;
import fl.events.ColorPickerEvent;
import fl.controls.RadioButtonGroup;

var rbGrp:RadioButtonGroup = RadioButtonGroup.getGroup("fontRbGrp");
rbGrp.addEventListener(MouseEvent.CLICK, rbHandler);
txtCp.addEventListener(ColorPickerEvent.CHANGE,cpHandler);
msgCb.addEventListener(Event.CHANGE, cbHandler);
```

처음 세 행은 응용 프로그램에서 사용하는 이벤트 클래스를 가져옵니다. 따라서 사용자가 구성 요소 중 하나와 상호 작용할 경우 이벤트가 발생합니다. 다음 다섯 행은 응용 프로그램에서 수신하려고 하는 이벤트에 대한 이벤트 핸들러를 등록합니다. 따라서 RadioButton을 클릭하면 이에 대한 click 이벤트가 발생하고 ColorPicker에서 다른 색상을 선택하면 change 이벤트가 발생합니다. 드롭 다운 목록에서 다른 인사말을 선택하면 ComboBox에서 change 이벤트가 발생합니다.

네 번째 행은 응용 프로그램이 리스너를 각 버튼에 개별적으로 할당하는 대신 이벤트 리스너를 RadioButton 그룹에 할당할 수 있도록 RadioButtonGroup 클래스를 가져옵니다.

11 [액션] 패널에 다음 코드 행을 추가하여 TextArea에서 텍스트의 size 및 color 스타일 속성을 변경할 때 응용 프로그램이 사용하는 tf TextFormat 객체를 만듭니다.

```
var tf:TextFormat = new TextFormat();
```

12 다음 코드를 추가하여 rbHandler 이벤트 처리 함수를 만듭니다. 이 함수는 사용자가 RadioButton 구성 요소 중 하나를 클릭할 때 click 이벤트를 처리합니다.

```
function rbHandler(event:MouseEvent):void {
    switch(event.target.selection.name) {
        case "smallRb":
            tf.size = 14;
            break;
        case "largerRb":
            tf.size = 18;
            break;
        case "largestRb":
            tf.size = 24;
            break;
        }
        aTa.setStyle("textFormat", tf);
}
```

이 함수는 switch 문으로 event 객체의 target 속성을 검사하여 어느 RadioButton이 해당 이벤트를 트리거했는지 확인합니다. currentTarget 속성에는 이벤트를 트리거한 객체의 이름이 포함되어 있습니다. 사용자가 클릭한 RadioButton에 따라 응용 프로그램은 TextArea에서 텍스트의 크기를 14, 18 또는 24포인트로 변경합니다.

13 다음 코드를 추가하여 ColorPicker의 값 변경 사항을 처리하는 cpHandler() 함수를 구현합니다.

```
function cpHandler(event:ColorPickerEvent):void {
    tf.color = event.target.selectedColor;
    aTa.setStyle("textFormat", tf);
}
```

이 함수는 tf TextFormat 객체의 color 속성을 ColorPicker에서 선택된 색상으로 설정하고 setStyle()을 호출하여 aTa TextArea 인스턴스의 텍스트에 해당 스타일을 적용합니다.

14 다음 코드를 추가하여 ComboBox 선택의 변경 내용을 처리하는 cbHandler() 함수를 구현합니다.

```
function cbHandler(event:Event):void {
    aTa.text = event.target.selectedItem.label;
}
```

이 함수는 단순히 TextArea의 텍스트를 ComboBox에서 선택된 텍스트인 event.target.selectedItem.label로 바꿉니다.

15 [컨트롤] > [동영상 테스트]를 선택하거나 Control+Enter를 눌러 코드를 컴파일하고 Greetings 응용 프로그램을 테스트합니다.

다음 단원에서는 외부 ActionScript 클래스 및 라이브러리에 필수 구성 요소만 있는 FLA 파일을 사용하여 같은 응용 프로그램을 만드는 방법을 보여 줍니다.

외부 클래스 파일을 사용하여 Greetings2 응용 프로그램 만들기:

1 [파일] > [새로 만들기]를 선택합니다.

2 [새 문서] 대화 상자에서 [Flash 파일(ActionScript 3.0)]을 선택하고 [확인]을 클릭합니다.

새 Flash 윈도우가 열립니다.

3 [파일] > [저장]을 선택하고 Flash 파일의 이름을 **Greetings2.fla**로 지정한 후 [저장] 버튼을 클릭합니다.

4 다음과 같은 각 구성 요소를 [구성 요소] 패널에서 라이브러리로 드래그합니다.

- ColorPicker
- ComboBox
- RadioButton
- TextArea

이러한 에셋은 컴파일된 SWF 파일에 사용되므로 라이브러리에 추가해야 합니다. 구성 요소를 [라이브러리] 패널 아래쪽으로 드래그합니다. 이러한 구성 요소를 라이브러리에 추가할 때 List, TextInput, UIScrollBox 등의 다른 에셋도 자동으로 추가됩니다.

5 [속성] 윈도우에서 [문서 클래스]에 **Greetings2**를 입력합니다.

문서 클래스에 대한 정의를 찾을 수 없다는 경고가 표시되면 무시합니다. Greetings2 클래스는 다음 단계에서 정의합니다. 이 클래스는 응용 프로그램의 기본 기능을 정의합니다.

6 Greetings2.fla 파일을 저장합니다.

7 [파일] > [새로 만들기]를 선택합니다.

8 [새 문서] 대화 상자에서 [ActionScript 파일]을 선택하고 [확인]을 클릭합니다.

새 스크립트 윈도우가 열립니다.

9 스크립트 윈도우에 다음 코드를 추가합니다.

```
package {
    import flash.display.Sprite;
    import flash.events.Event;
    import flash.events.MouseEvent;
    import flash.text.TextFormat;
    import fl.events.ComponentEvent;
    import fl.events.ColorPickerEvent;
    import fl.controls.ColorPicker;
    import fl.controls.ComboBox;
    import fl.controls.RadioButtonGroup;
    import fl.controls.RadioButton;
    import fl.controls.TextArea;
    public class Greetings2 extends Sprite {
        private var aTa:TextArea;
        private var msgCb:ComboBox;
        private var smallRb:RadioButton;
        private var largerRb:RadioButton;
        private var largestRb:RadioButton;
        private var rbGrp:RadioButtonGroup;
        private var txtCp:ColorPicker;
        private var tf:TextFormat = new TextFormat();
        public function Greetings2() {
```

이 스크립트는 Greetings2라는 ActionScript 3.0 클래스를 정의하며 다음과 같은 작업을 수행합니다.

- 파일에 사용될 클래스를 가져옵니다. 일반적으로 코드에서 여러 클래스를 참조할 때 이러한 import 문을 추가하게 되지만 이 예제에서는 간단히 하기 위해 모두 하나의 단계로 가져옵니다.
- 코드에 추가할 여러 유형의 구성 요소 객체를 나타내는 변수를 선언합니다. 다른 변수는 tf TextFormat 객체를 만듭니다.
- 클래스에 대해 생성자 함수 Greetings2()를 정의합니다. 이 단계에서는 이 함수에 행을 추가하고 다음 단계에서는 클래스에 다른 메서드를 추가합니다.

10 [파일] > [저장]을 선택하고 파일의 이름을 **Greetings2.as**로 지정한 후 [저장] 버튼을 클릭합니다.

11 Greeting2() 함수에 다음 코드 행을 추가합니다.

```
        createUI();
        setUpHandlers();
    }
```

함수는 이제 다음과 같아야 합니다.

```
    public function Greetings2() {
        createUI();
        setUpHandlers();
    }
```

12 Greeting2() 메서드의 닫는 중괄호 뒤에 다음 코드 행을 추가합니다.

```
private function createUI() {
    bldTxtArea();
    bldColorPicker();
    bldComboBox();
    bldRadioButtons();
}
private function bldTxtArea() {
    aTa = new TextArea();
    aTa.setSize(230, 44);
    aTa.text = "Hello World!";
    aTa.move(165, 57);
    addChild(aTa);
}
private function bldColorPicker() {
    txtCp = new ColorPicker();
    txtCp.move(96, 72);
    addChild(txtCp);
}
private function bldComboBox() {
    msgCb = new ComboBox();
    msgCb.width = 130;
    msgCb.move(265, 120);
    msgCb.prompt = "Greetings";
    msgCb.addItem({data:"Hello.", label:"English"});
    msgCb.addItem({data:"Bonjour.", label:"Français"});
    msgCb.addItem({data:"¡Hola!", label:"Español"});
    addChild(msgCb);
}
private function bldRadioButtons() {
    rbGrp = new RadioButtonGroup("fontRbGrp");
    smallRb = new RadioButton();
    smallRb.setSize(100, 22);
    smallRb.move(155, 120);
    smallRb.group = rbGrp; //"fontRbGrp";
    smallRb.label = "Small";
    smallRb.name = "smallRb";
    addChild(smallRb);
    largerRb = new RadioButton();
    largerRb.setSize(100, 22);
    largerRb.move(155, 148);
    largerRb.group = rbGrp;
    largerRb.label = "Larger";
    largerRb.name = "largerRb";
    addChild(largerRb);
    largestRb = new RadioButton();
    largestRb.setSize(100, 22);
    largestRb.move(155, 175);
    largestRb.group = rbGrp;
    largestRb.label = "Largest";
    largestRb.name = "largestRb";
    addChild(largestRb);
}
```

이 행은 다음과 같은 작업을 수행합니다.

- 응용 프로그램에 사용되는 구성 요소를 인스턴스화합니다.
- 각 구성 요소의 크기, 위치 및 속성을 설정합니다.
- addChild() 메서드를 사용하여 각 구성 요소를 스테이지에 추가합니다.

13 bldRadioButtons() 메서드의 닫는 중괄호 뒤에 setUpHandlers() 메서드에 대한 다음 코드를 추가합니다.

```
private function setUpHandlers():void {
    rbGrp.addEventListener(MouseEvent.CLICK, rbHandler);
    txtCp.addEventListener(ColorPickerEvent.CHANGE,cpHandler);
    msgCb.addEventListener(Event.CHANGE, cbHandler);
}
private function rbHandler(event:MouseEvent):void {
    switch(event.target.selection.name) {
        case "smallRb":
            tf.size = 14;
            break;
        case "largerRb":
            tf.size = 18;
            break;
        case "largestRb":
            tf.size = 24;
            break;
    }
    aTa.setStyle("textFormat", tf);
}
private function cpHandler(event:ColorPickerEvent):void {
    tf.color = event.target.selectedColor;
    aTa.setStyle("textFormat", tf);
}
private function cbHandler(event:Event):void {
    aTa.text = event.target.selectedItem.data;
}
}
}
```

이러한 함수는 구성 요소에 대한 이벤트 리스너를 정의합니다.

14 [파일] > [저장]을 선택하여 파일을 저장합니다.

15 [컨트롤] > [동영상 테스트]를 선택하거나 Control+Enter를 눌러 코드를 컴파일하고 Greetings2 응용 프로그램을 테스트합니다.

후속 예제 개발 및 실행

Greetings 응용 프로그램을 개발하고 실행한 후에는 이 설명서에 있는 다른 코드 예제를 실행하는 데 필요한 기본적인 지식을 익혀야 합니다. 각 예제에서는 관련 ActionScript 3.0 코드가 강조 표시되고 설명되어 있으므로 이 설명서의 각 예제를 잘라내 FLA 파일에 붙여 넣은 다음 컴파일하고 실행할 수 있습니다.

3장: 구성 요소를 사용한 작업

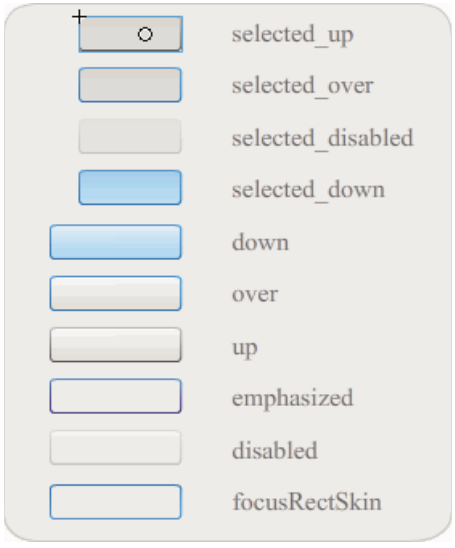
구성 요소 아키텍처

Adobe® ActionScript® 3.0 구성 요소는 Adobe® Flash Player 버전 9.0.28.0 이상에서만 지원되며 Flash CS4 이전에 만들어진 구성 요소와는 호환되지 않습니다. Adobe® ActionScript® 2.0 구성 요소 사용에 대한 자세한 내용은 **Adobe® ActionScript® 2.0 구성 요소 사용 설명서** 및 **Adobe® ActionScript® 2.0 구성 요소 언어 참조 설명서**를 참조하십시오.

Adobe ActionScript 3.0 UI(사용자 인터페이스) 구성 요소는 FLA 기반 구성 요소로 구현되지만 Flash CS5에서는 SWC 및 FLA 기반 구성 요소를 모두 지원합니다. 예를 들어, FLVPlayback 및 FLVPlaybackCaptioning 구성 요소는 SWC 기반 구성 요소입니다. 이 두 가지 유형의 구성 요소 중 어느 것이든 Components 폴더에 추가하면 해당 구성 요소가 [구성 요소] 패널에 표시됩니다. 이러한 두 구성 요소는 서로 다르게 만들어졌으므로 개별적으로 살펴보겠습니다.

ActionScript 3.0 FLA 기반 구성 요소

ActionScript 3.0 사용자 인터페이스 구성 요소는 FLA 기반 파일(.fla)이며 스테이지에서 구성 요소를 두 번 클릭하여 편집할 수 있는 내장 스킨이 이 구성 요소에 포함되어 있습니다. 구성 요소의 스킨과 기타 에셋은 타임라인의 프레임 2에 있습니다. 구성 요소를 두 번 클릭하면 자동으로 프레임 2로 이동되며 구성 요소의 스킨 팔레트가 열립니다. 다음 그림에서는 Button 구성 요소에 대해 표시되는 스킨 팔레트를 보여 줍니다.



Button 구성 요소의 스킨

구성 요소 스킨 및 구성 요소 사용자 정의에 대한 자세한 내용은 89페이지의 "UI 구성 요소 사용자 정의" 및 138페이지의 "FLVPlayback 구성 요소 사용자 정의"를 참조하십시오.

Flash CS5 FLA 기반 UI 구성 요소에는 구성 요소의 미리 컴파일된 ActionScript 코드가 들어 있는 SWC도 포함되어 있어 응용 프로그램의 컴파일 속도를 크게 향상시키고 ActionScript 3.0 설정과 충돌하는 것을 방지합니다. 미리 컴파일된 정의를 사용할 수 있도록 ComponentShim SWC는 스테이지에서 모든 사용자 인터페이스 구성 요소의 프레임 2에 있습니다. 스테이지에 있는 구성 요소나 해당 [링크] 속성에서 [첫 프레임으로 내보내기] 옵션이 선택된 상태로 라이브러리에 있는 구성 요소만 ActionScript에 사용할 수 있습니다. ActionScript를 사용하여 구성 요소를 만들려면 import 문을 통해 클래스를 가져와서 구성 요소에 액세스해야 합니다. import 문에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**을 참조하십시오.

SWC 기반 구성 요소

SWC 기반 구성 요소에도 FLA 파일과 ActionScript 클래스 파일이 있지만 해당 파일은 SWC로 컴파일하고 내보낸 것입니다. 미리 컴파일된 Flash 심볼과 ActionScript 코드의 패키지인 SWC 파일을 사용하면 변경되지 않는 심볼과 코드를 다시 컴파일하지 않아도 됩니다.

FLVPlayback 및 FLVPlaybackCaptioning 구성 요소는 SWC 기반 구성 요소입니다. 이러한 구성 요소의 스킨은 기본 제공 스킨이 아닌 외부 스킨입니다. FLVPlayback 구성 요소에는 기본 스킨이 있습니다. 기본 스킨은 미리 제작된 스킨 컬렉션에서 하나를 선택하거나, [구성 요소] 패널의 UI 컨트롤에서 컨트롤을 사용자 정의하거나(BackButton, BufferingBar 등), 사용자 정의 스킨을 만드는 방법으로 변경할 수 있습니다. 자세한 내용은 138페이지의 "FLVPlayback 구성 요소 사용자 정의"를 참조하십시오.

Flash에서는 다음과 같은 방법으로 무비 클립을 컴파일된 클립으로 변환할 수 있습니다.

무비 클립 컴파일

- [라이브러리] 패널에서 무비 클립을 마우스 오른쪽 버튼으로 클릭(Windows)하거나 Control 키를 누른 채 클릭(Macintosh)하고 [컴파일된 클립으로 변환]을 선택합니다.

  컴파일된 클립은 컴파일하기 전의 무비 클립과 동일하게 동작하지만 일반 무비 클립보다 더 빠르게 표시되고 제작됩니다. 컴파일된 클립은 편집할 수 없지만 속성 관리자와 [구성 요소 관리자]에 해당 속성이 나타납니다.

  SWC 구성 요소에는 컴파일된 클립, 구성 요소의 미리 컴파일된 ActionScript 정의 및 구성 요소를 설명하는 기타 파일이 포함되어 있습니다. 구성 요소를 직접 만든 경우에는 SWC 파일로 내보내서 배포할 수 있습니다.

SWC 파일 내보내기

- [라이브러리] 패널에서 무비 클립을 선택하고 마우스 오른쪽 버튼으로 클릭(Windows)하거나 Control 키를 누른 상태에서 클릭(Macintosh)한 다음 [SWC 파일 내보내기]를 선택합니다.

  참고: Flash CS4 이상의 SWC 파일 형식은 Flex SWC 형식과 호환되기 때문에 이 두 제품 간에 SWC 파일을 교환할 수 있지만 어느 정도의 수정은 필요할 수 있습니다.

SWC 기반 구성 요소 만들기에 대한 자세한 내용은 www.adobe.com/go/learn_fl_creating_components_kr을 참조하십시오.

ActionScript 3.0 구성 요소 API

ActionScript 3.0 구성 요소 각각은 패키지 폴더에서 이름이 **fl.packagename.classname** 형식인 ActionScript 3.0 클래스를 기반으로 만들어집니다. 예를 들어, Button 구성 요소는 Button 클래스의 인스턴스이며 패키지 이름은 fl.controls.Button입니다. 응용 프로그램에 구성 요소 클래스를 가져올 때는 해당 패키지 이름을 반드시 참조해야 합니다. Button 클래스의 경우에는 다음과 같은 명령문을 사용하여 가져올 수 있습니다.

```
import fl.controls.Button;
```

구성 요소 클래스 파일의 위치에 대한 자세한 내용은 17페이지의 "구성 요소 파일을 사용한 작업"을 참조하십시오.

구성 요소의 클래스는 응용 프로그램에서 해당 구성 요소와 상호 작용하는 데 사용할 수 있는 메서드, 속성, 이벤트 및 스타일을 정의합니다. ActionScript 3.0 UI 구성 요소는 Sprite 및 UIComponent 클래스의 하위 클래스이며 이들 클래스의 속성, 메서드 및 이벤트를 상속합니다. Sprite 클래스는 기본 표시 목록을 구성하는 단위로, 타임라인이 없다는 점을 제외하면 MovieClip 클래스와 비슷합니다. UIComponent 클래스는 모든 대화형 및 비대화형 시각적 구성 요소의 기본 클래스입니다. 각 구성 요소의 상속 경로와 해당 속성, 메서드, 이벤트 및 스타일은 **Adobe ActionScript 3.0 Reference for Flash Professional**에 설명되어 있습니다.

모든 ActionScript 3.0 구성 요소는 ActionScript 3.0 이벤트 처리 모델을 사용합니다. 이벤트 처리에 대한 자세한 내용은 21페이지의 "이벤트 처리" 및 **ActionScript 3.0** 프로그래밍을 참조하십시오.

구성 요소 파일을 사용한 작업

이 단원에서는 구성 요소 파일이 저장되는 위치, ActionScript 소스 파일의 위치 및 [구성 요소] 패널에서 구성 요소를 추가하고 제거하는 방법에 대해 설명합니다.

구성 요소 파일이 저장되는 위치

Flash 구성 요소는 응용 프로그램 수준의 Configuration 폴더에 저장됩니다.

참고: 이 폴더에 대한 자세한 내용은 Flash 사용 설명서의 "Flash와 함께 설치되는 Configuration 폴더"를 참조하십시오.

구성 요소는 다음 위치에 설치됩니다.

- Windows 2000 또는 Windows XP: C:\Program Files\Adobe\Adobe Flash CS5**language**\Configuration\Components
- Mac OS X: Macintosh HD:Applications:Adobe Flash CS5:Configuration:Components

  Components 폴더에서 UI(사용자 인터페이스) 구성 요소는 User Interface.fla 파일에 저장되고 FLVPlayback(FLVPlaybackAS3.swc) 및 FLVPlaybackCaptioning 구성 요소는 Video 폴더에 저장됩니다.

  다음과 같이 사용자가 지정한 위치에 구성 요소를 저장할 수도 있습니다.

- Windows 2000 또는 Windows XP: C:\Documents and Settings**username**\Local Settings\Application Data\Adobe\Adobe Flash CS5\ko\Configuration\Components
- Windows Vista: C:\Users**username**\Local Settings\Application Data\Adobe\Adobe Flash CS5\ko\Configuration\Components

  참고: Windows에서 Application Data 폴더는 기본적으로 숨겨져 있습니다. 숨겨진 폴더와 파일을 표시하려면 [내 컴퓨터]를 선택하여 Windows 탐색기를 열고, [도구] > [폴더 옵션]을 선택한 다음 [보기] 탭을 선택하십시오. [보기] 탭에서 [숨김 파일 및 폴더 표시] 라디오 버튼을 선택합니다.

- Mac OS X: Macintosh HD:Users:<username>:Library:Application Support:Adobe Flash CS5:Configuration:Components

구성 요소 소스 파일이 저장되는 위치

Windows 2000 또는 Windows XP의 경우 구성 요소의 ActionScript 클래스 파일(.as) 또는 소스 파일은 다음 응용 프로그램 폴더에 설치됩니다.

사용자 인터페이스 구성 요소 C:\Program Files\Adobe\Adobe Flash CS5\ko\Configuration\Component Source\ActionScript 3.0\User Interface\fl

FLVPlayback C:\Program Files\Adobe\Adobe Flash CS5\ko\Configuration\Component Source\ActionScript 3.0\FLVPlayback\fl\video

FLVPlaybackCaptioning C:\Program Files\Adobe\Adobe Flash CS5\ko\Configuration\Component Source\ActionScript 3.0\FLVPlaybackCaptioning\fl\video

Mac OS X의 경우 구성 요소 소스 파일은 다음 위치에 저장됩니다.

사용자 인터페이스 구성 요소 Macintosh HD:Applications:Adobe Flash CS5:Configuration:Component Source:ActionScript 3.0:User Interface:fl

FLVPlayback Macintosh HD:Applications:Adobe Flash CS5:Configuration:Component Source:ActionScript 3.0:FLVPlayback:fl:video

FLVPlaybackCaptioning Macintosh HD:Applications:Adobe Flash CS5:Configuration:Component Source:ActionScript 3.0:FLVPlaybackCaptioning:fl:video

구성 요소 소스 파일 및 클래스 경로

ActionScript 3.0 구성 요소는 코드가 컴파일되어 있으므로 ActionScript 클래스 파일의 위치를 클래스 경로 변수에 지정하면 안 됩니다. 클래스 경로에 클래스 파일 위치를 포함하면 응용 프로그램을 컴파일하는 데 시간이 더 오래 걸립니다. 그러나 클래스 경로 설정에 구성 요소 클래스 파일이 있는 경우 클래스 파일이 구성 요소의 컴파일된 코드보다 항상 우선합니다.

구성 요소가 포함된 응용 프로그램을 디버깅할 때는 클래스 경로 설정에 구성 요소 소스 파일의 위치를 추가할 수도 있습니다. 자세한 내용은 18페이지의 "구성 요소 응용 프로그램 디버깅"을 참조하십시오.

구성 요소 파일 수정

Flash에 SWC 기반 구성 요소를 업데이트, 추가 또는 제거하거나 새로운 FLA 기반 구성 요소를 추가한 후에는 구성 요소를 [구성 요소] 패널에서 다시 로드해야 사용할 수 있습니다. 구성 요소를 다시 로드하려면 Flash를 다시 시작하거나 [구성 요소] 패널 메뉴에서 [다시 로드]를 선택합니다. 이렇게 하면 Components 폴더에 추가한 구성 요소가 인식됩니다.

Flash가 실행 중일 때 구성 요소 패널에서 구성 요소 다시 로드:

- [구성 요소] 패널 메뉴에서 [다시 로드]를 선택합니다.

구성 요소 패널에서 구성 요소 제거:

- Components 폴더에서 FLA, SWC 또는 MXP 파일을 제거한 후 Flash를 다시 시작하거나 [구성 요소] 패널 메뉴에서 [다시 로드]를 선택합니다. MXP 파일은 Adobe Exchange에서 다운로드한 구성 요소 파일입니다.

  Flash가 실행 중일 때 SWC 기반 구성 요소를 제거하거나 바꿀 수 있습니다. 이러한 경우 구성 요소를 다시 로드하면 변경 내용이 적용되지만 FLA 기반 구성 요소를 변경하거나 삭제한 경우에는 Flash를 종료한 후 다시 시작해야만 변경 내용이 적용됩니다. 그러나 FLA 기반 구성 요소를 추가한 후 [다시 로드] 명령을 사용하여 해당 구성 요소를 로드할 수 있습니다.

  Flash 구성 요소 파일(.fla 또는 .as)을 변경할 때는 먼저 해당 파일의 복사본을 만들어 두는 것이 좋습니다. 이렇게 하면 필요 시 파일을 복원할 수 있습니다.

구성 요소 응용 프로그램 디버깅

응용 프로그램을 컴파일할 때 소요되는 컴파일 시간을 줄이기 위해 ActionScript 3.0 구성 요소에는 해당 구성 요소의 모든 소스 코드가 들어 있습니다. 그러나 Flash 디버거는 컴파일된 클립 안의 코드를 검사할 수 없기 때문에 응용 프로그램을 디버깅할 때 구성 요소의 소스 코드까지 디버깅하려면 클래스 경로 설정에 구성 요소 소스 파일을 추가해야 합니다.

구성 요소 패키지 폴더의 위치는 해당 구성 요소 유형의 소스 파일이 있는 위치에 따라 다릅니다. 모든 UI 구성 요소에 대해 ActionScript 3.0 소스 파일을 모두 참조하려면 사용자 인터페이스 패키지의 클래스 경로에 다음 위치를 추가합니다.

- $(AppConfig)/Component Source/ActionScript 3.0/User Interface

참고: 이렇게 하면 모든 UI 구성 요소의 컴파일된 코드가 무시되고 응용 프로그램 컴파일 시간이 길어집니다. 어떤 이유로든 구성 요소의 소스 파일을 변경하면 해당 구성 요소의 비헤이비어가 다르게 나타날 수 있습니다.

클래스 경로를 설정하려면 [편집] 메뉴에서 [환경 설정]을 선택하고 [범주] 목록에서 [ActionScript]를 선택한 후 [ActionScript 3.0 설정 버튼을 클릭합니다. 새 항목을 추가하려면 현재 설정을 표시하는 윈도우 위쪽에 있는 더하기 기호를 클릭합니다.

$(AppConfig) 변수는 Flash CS5을 설치한 위치의 Flash CS5 Configuration 폴더를 가리킵니다. 일반적으로 경로는 다음과 같습니다.

- Windows 2000 또는 Windows XP: C:\Program Files\Adobe\Adobe Flash CS5\language\Configuration\
- Mac OS X: Macintosh HD:Applications:Adobe Flash CS5:Configuration

참고: 구성 요소 소스 파일을 변경해야 할 경우에는 원본 소스 파일을 다른 위치에 복사한 후 이 위치를 클래스 경로에 추가하는 것이 좋습니다.

구성 요소 소스 파일의 위치에 대한 자세한 내용은 17페이지의 "구성 요소 소스 파일이 저장되는 위치"를 참조하십시오.

매개 변수 및 속성 설정

각 구성 요소에는 모양과 비헤이비어를 변경하는 데 사용할 수 있는 매개 변수가 있습니다. 매개 변수는 구성 요소 클래스의 속성이며 속성 관리자와 [구성 요소 관리자]에 표시됩니다. 가장 일반적으로 사용되는 속성은 제작 매개 변수로 표시되지만 그 외 다른 속성은 ActionScript를 사용하여 설정해야 합니다. 제작하는 동안에 설정할 수 있는 모든 매개 변수를 ActionScript로 설정할 수도 있습니다. ActionScript로 매개 변수를 설정하면 제작하는 동안에 설정된 모든 값이 재정의됩니다.

대부분의 ActionScript 3.0 사용자 인터페이스 구성 요소는 UIComponent 클래스 및 기본 클래스의 속성과 메서드를 상속합니다. 예를 들어, Button 및 CheckBox 클래스는 UIComponent 클래스 및 BaseButton 클래스 둘 모두의 속성을 상속합니다. 사용자는 구성 요소의 상속된 속성과 고유한 클래스 속성에 모두 액세스할 수 있습니다. 예를 들어, ProgressBar 구성 요소에는 UIComponent에서 상속한 ProgressBar.enabled 속성뿐만 아니라 고유한 ProgressBar.percentComplete 속성도 있으며 이 두 가지 속성을 모두 사용하여 ProgressBar 구성 요소의 인스턴스와 상호 작용할 수 있습니다. 구성 요소의 속성에 대한 자세한 내용은 ActionScript 3.0 참조 설명서를 참조하십시오.

속성 관리자나 [구성 요소 관리자]를 사용하여 구성 요소 인스턴스의 매개 변수를 설정할 수 있습니다.

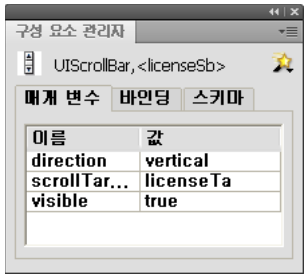
속성 관리자에서 구성 요소의 인스턴스 이름 입력:

1. [윈도우] > [속성] > [속성]을 선택합니다.
2. 스테이지에서 구성 요소 인스턴스를 선택합니다.
3. [무비 클립] 드롭 다운 목록 아래쪽에 있는 [<인스턴스 이름>] 텍스트 상자에 구성 요소 인스턴스의 이름을 입력합니다. 또는 [매개 변수] 탭을 클릭하고 **구성 요소**라는 단어 아래의 상자에 이름을 입력합니다. 그런 다음 설정할 매개 변수의 값을 입력합니다.

   인스턴스 이름에 구성 요소의 종류를 나타내는 접미어를 추가하는 것이 좋습니다. 이렇게 하면 ActionScript 코드를 보다 쉽게 읽을 수 있습니다. 예를 들어, **licenseSb**라는 인스턴스 이름은 구성 요소가 **licenseTa** 텍스트 영역에서 사용권 계약을 스크롤하는 스크롤 막대임을 나타냅니다.

구성 요소 관리자에서 구성 요소 인스턴스의 매개 변수 입력:

1. [윈도우] > [구성 요소 관리자]를 선택합니다.
2. 스테이지에서 구성 요소 인스턴스를 선택합니다.
3. [매개 변수] 탭을 클릭한 다음 나열된 매개 변수의 값을 입력합니다.



구성 요소 관리자의 구성 요소 매개 변수

ActionScript의 구성 요소 속성 설정

ActionScript에서 도트(.) 연산자(도트 구문)는 스테이지의 객체 또는 인스턴스에 속한 속성이나 메서드에 액세스하는 데 사용됩니다. 도트 구문 표현식은 인스턴스의 이름으로 시작하고 그 뒤에 도트가 있으며 지정할 요소가 맨 마지막에 나옵니다. 예를 들어, 다음 ActionScript 코드는 CheckBox 인스턴스 aCh의 width 속성을 50픽셀로 설정합니다.

```
aCh.width = 50;
```

다음 if 문은 사용자가 체크 상자를 선택했는지 여부를 확인합니다.

```
if (aCh.selected == true) {
    displayImg(redCar);
}
```

라이브러리

문서에 구성 요소를 처음 추가할 때 Flash에서 해당 구성 요소를 라이브러리 패널에 무비 클립으로 가져옵니다. 구성 요소 패널에서 라이브러리 패널로 구성 요소를 직접 드래그한 후 스테이지에 해당 구성 요소의 인스턴스를 추가할 수도 있습니다. 어느 방법을 사용하든 구성 요소의 클래스 요소에 액세스하려면 먼저 구성 요소를 라이브러리에 추가해야 합니다.

라이브러리에 구성 요소를 추가한 후 ActionScript를 사용하여 구성 요소의 인스턴스를 만들려면 먼저 import 문을 사용하여 해당 클래스를 가져와야 합니다. import 문에서 구성 요소의 패키지 이름과 클래스 이름을 모두 지정해야 합니다. 예를 들어, 다음 명령문은 Button 클래스를 가져옵니다.

```
import fl.controls.Button;
```

구성 요소를 라이브러리에 추가하면 Flash에서는 구성 요소의 여러 상태에 사용할 스킨이 포함된 에셋 폴더를 함께 가져옵니다. 구성 요소의 **스킨**은 응용 프로그램에서 구성 요소의 그래픽 표시를 구성하는 심볼 컬렉션으로 이루어져 있습니다. 스킨 하나는 구성 요소의 특정한 상태를 나타내는 그래픽 표현 또는 무비 클립입니다.

필요한 경우 Component Assets 폴더 안의 항목을 사용하여 구성 요소의 스킨을 변경할 수 있습니다. 자세한 내용은 89페이지의 “UI 구성 요소 사용자 정의”를 참조하십시오.

구성 요소를 라이브러리에 추가한 후에는 [구성 요소] 패널이나 [라이브러리] 패널에서 스테이지로 해당 구성 요소의 아이콘을 드래그하여 문서에 구성 요소의 인스턴스를 여러 개 만들 수 있습니다.

구성 요소 크기 조절

[자유 변형 도구]나 setSize() 메서드를 사용하면 구성 요소 인스턴스의 크기를 조절할 수 있습니다. 모든 구성 요소 인스턴스에서 setSize() 메서드를 호출하여 크기를 조절할 수 있습니다(UIComponent.setSize() 참조). 다음 코드는 List 구성 요소 인스턴스의 크기를 200픽셀(폭) x 300픽셀(높이)로 조절합니다.

```
aList.setSize(200, 300);
```

구성 요소는 레이블에 맞게 자동으로 크기가 조절되지 않습니다. 문서에 추가한 구성 요소 인스턴스가 작아서 레이블을 표시할 수 없으면 레이블 텍스트가 잘립니다. 사용자가 레이블에 맞게 구성 요소 크기를 조절해야 합니다.

구성 요소 크기 조절에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 해당 개별 항목을 참조하십시오.

실시간 미리 보기

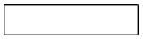
기본적으로 활성화되어 있는 실시간 미리 보기 기능을 사용하면 제작된 Flash 내용에 구성 요소가 표시되는 모양을 스테이지에서 실제 크기로 미리 볼 수 있습니다.

실시간 미리 보기를 활성화하거나 비활성화하려면:

- [컨트롤] > [실시간 미리 보기 활성화]를 선택합니다. 이 옵션 옆에 있는 체크 상자가 선택되어 있으면 실시간 미리 보기가 활성화되어 있다는 것을 나타냅니다.

실시간 미리 보기에 반영되는 매개 변수는 구성 요소마다 다릅니다. 실시간 미리 보기에 반영되는 구성 요소 매개 변수에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 각 구성 요소 항목을 참조하십시오.

실시간 미리 보기가 활성화된 Button 구성 요소



실시간 미리 보기가 비활성화된 Button 구성 요소

실시간 미리 보기 상태에서는 구성 요소가 작동하지 않습니다. 기능을 테스트하려면 [컨트롤] > [무비 테스트] 명령을 사용해야 합니다.

이벤트 처리

모든 구성 요소는 사용자가 구성 요소와 상호 작용할 때 이벤트를 브로드캐스팅합니다. 예를 들어, 사용자가 Button을 클릭하면 Button 구성 요소는 MouseEvent.CLICK 이벤트를 전달하고, 사용자가 List에서 항목을 선택하면 List 구성 요소는 Event.CHANGE 이벤트를 전달합니다. UILoader 인스턴스의 내용 로드가 완료되어 Event.COMPLETE 이벤트가 생성되는 경우와 같이 구성 요소에 중요한 변화가 나타날 때도 이벤트가 발생할 수 있습니다. 이벤트를 처리하려면 이벤트가 발생할 때 실행되는 ActionScript 코드를 작성해야 합니다.

구성 요소의 이벤트에는 구성 요소가 상속받는 모든 클래스의 이벤트가 포함됩니다. 즉, UIComponent 클래스는 ActionScript 3.0 사용자 인터페이스 구성 요소의 기본 클래스이기 때문에 모든 ActionScript 3.0 사용자 인터페이스 구성 요소는 이 클래스의 이벤트를 상속합니다. 구성 요소가 브로드캐스트하는 이벤트 목록은 **ActionScript 3.0 Reference for Flash Professional**에서 해당 구성 요소 클래스 항목의 이벤트 단원을 참조하십시오.

ActionScript 3.0에서의 이벤트 처리에 대한 자세한 설명을 보려면 **ActionScript 3.0** 프로그래밍을 참조하십시오.

이벤트 리스너

다음은 ActionScript 3.0 구성 요소의 이벤트 처리와 관련된 주요 사항입니다.

- 구성 요소 클래스의 인스턴스가 모든 이벤트를 브로드캐스팅합니다. 즉, 구성 요소 인스턴스가 **브로드캐스터**입니다.
- 이벤트 **리스너**를 등록하려면 구성 요소 인스턴스에 대해 addEventListener() 메서드를 호출해야 합니다. 예를 들어, 다음 코드 행은 Button 인스턴스 aButton에 MouseEvent.CLICK 이벤트에 대한 리스너를 추가합니다.

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler);
  ```

  addEventListener() 메서드의 두 번째 매개 변수는 이벤트가 발생할 때 호출할 함수 이름인 clickHandler를 등록합니다. 이 함수는 **콜백함수**라고도 합니다.

- 구성 요소 인스턴스 하나에 리스너를 여러 개 등록할 수 있습니다.

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  aButton.addEventListener(MouseEvent.CLICK, clickHandler2);
  ```

- 하나의 리스너를 여러 구성 요소 인스턴스에 등록할 수 있습니다.

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
bButton.addEventListener(MouseEvent.CLICK, clickHandler1);
```

- 이벤트 핸들러 함수는 이벤트 유형 및 해당 이벤트를 브로드캐스팅하는 인스턴스에 대한 정보가 들어 있는 이벤트 객체로 전달됩니다. 자세한 내용은 22페이지의 “이벤트 객체”를 참조하십시오.
- 리스너는 응용 프로그램이 종료되거나 사용자가 removeEventListener() 메서드를 사용하여 리스너를 명시적으로 제거하기 전까지 활성 상태로 유지됩니다. 예를 들어, 다음 코드 행은 aButton의 MouseEvent.CLICK 이벤트에 대한 리스너를 제거합니다.

```
aButton.removeEventListener(MouseEvent.CLICK, clickHandler);
```

이벤트 객체

이벤트 객체는 Event 객체 클래스에서 상속되며 이벤트 객체에는 발생한 이벤트에 대한 중요한 정보를 제공하는 target 속성과 type 속성 등의 이벤트 정보가 포함된 속성이 있습니다.

속성	설명
type	이벤트 유형을 나타내는 문자열입니다.
target	이벤트를 브로드캐스팅하는 구성 요소 인스턴스에 대한 참조입니다.

이벤트에 추가 속성이 있는 경우 **ActionScript 3.0 Reference for Flash Professional**에서 이벤트의 클래스 설명에 해당 속성이 나열됩니다.

이벤트가 발생하면 이벤트 객체가 자동으로 생성되어 이벤트 핸들러 함수에 전달됩니다.

함수 내에 이벤트 객체를 사용하여 브로드캐스팅된 이벤트의 이름이나 이벤트를 브로드캐스팅하는 구성 요소의 인스턴스 이름에 액세스할 수 있습니다. 인스턴스 이름에서는 다른 구성 요소 속성에 액세스할 수 있습니다. 예를 들어, 다음 코드는 evtObj 이벤트 객체의 target 속성을 사용하여 aButton 인스턴스의 label 속성에 액세스하고 해당 값을 출력 패널에 표시합니다.

```
import fl.controls.Button;
import flash.events.MouseEvent;

var aButton:Button = new Button();
aButton.label = "Submit";
addChild(aButton);
aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(evtObj:MouseEvent){
trace("The " + evtObj.target.label + " button was clicked");
}
```

표시 목록을 사용한 작업

모든 ActionScript 3.0 구성 요소는 DisplayObject 클래스에서 상속되기 때문에 이 클래스의 메서드와 속성에 액세스하여 표시 목록과 상호 작용할 수 있습니다. 표시 목록은 응용 프로그램에서 표시되는 객체와 시각적 요소가 계층 구조를 이룬 것입니다. 이 계층 구조에는 다음과 같은 요소가 있습니다.

- 최상위 컨테이너인 스테이지
- 모양, 무비 클립, 텍스트 필드 등의 표시 객체
- 자식 표시 객체를 포함할 수 있는 특수한 유형의 표시 객체인 표시 객체 컨테이너

표시 목록의 객체 순서에 따라 부모 컨테이너에서 해당 객체의 심도가 결정됩니다. 객체의 심도는 스테이지 또는 표시 컨테이너 안에서 위쪽에서 아래쪽 방향 또는 앞쪽에서 뒤쪽 방향을 기준으로 해당 객체의 위치를 나타냅니다. 심도 순서는 객체가 서로 겹쳐 있는 경우에는 잘 드러나지만 그렇지 않은 경우에도 여전히 존재합니다. 표시 목록에 포함된 모든 객체에는 스테이지에서 특정한 심도가 설정되어 있습니다. 객체를 다른 객체 앞에 배치하거나 뒤로 이동하여 심도를 변경하려면 표시 목록에서 해당 객체의 위치를 변경해야 합니다. 표시 목록에 있는 객체의 기본 순서는 스테이지에서 객체가 배치되는 순서를 나타냅니다. 즉, 표시 목록에서 위치가 0인 객체는 심도 순서에서 맨 아래쪽에 위치합니다.

표시 목록에 구성 요소 추가

DisplayObjectContainer의 addChild() 또는 addChildAt() 메서드를 호출하여 DisplayObjectContainer 객체에 객체를 추가할 수 있습니다. 스테이지에서는 제작하는 동안 객체를 만들어 표시 목록에 추가할 수 있으며, 구성 요소의 경우에는 [구성 요소] 패널에서 스테이지로 구성 요소를 드래그하여 표시 목록에 추가할 수 있습니다. ActionScript를 사용하여 컨테이너에 객체를 추가하려면 먼저 new 연산자와 함께 해당 객체의 생성자를 호출하여 객체의 인스턴스를 만든 다음 addChild() 또는 addChildAt() 메서드를 호출하여 스테이지와 표시 목록에 객체를 배치해야 합니다. addChild() 메서드는 표시 목록의 다음 위치에 객체를 추가하고 addChildAt()은 지정한 위치에 객체를 추가합니다. 이미 사용 중인 위치를 지정하면 해당 위치에 있는 객체와 그 위쪽에 있는 모든 객체가 1씩 위쪽으로 이동합니다. DisplayObjectContainer 객체의 numChildren 속성에는 포함된 표시 객체의 수가 들어 있습니다. 위치를 지정하여 getChildAt() 메서드를 호출하거나, 객체 이름을 알고 있는 경우 getChildByName() 메서드를 호출하여 표시 목록에서 객체를 검색할 수 있습니다.

참고: ActionScript를 사용하여 구성 요소를 추가할 때 표시 목록에서 이름을 사용하여 구성 요소에 액세스하려면 구성 요소의 이름 속성에 이름을 지정해야 합니다.

다음 예제에서는 표시 목록에 있는 세 가지 구성 요소의 이름과 위치를 보여 줍니다. 우선 NumericStepper, Button 및 ComboBox를 서로 겹치게 스테이지에 드래그한 후 인스턴스 이름을 **aNs**, **aButton** 및 **aCb**로 각각 지정합니다. 그런 다음, 다음 코드를 타임라인의 프레임 1에 있는 액션 패널에 추가합니다.

```
var i:int = 0;
while(i < numChildren) {
    trace(getChildAt(i).name + " is at position: " + i++);
}
```

[출력] 패널에 다음 행이 표시됩니다.

```
aNs is at position: 0
aButton is at position: 1
aCb is at position: 2
```

표시 목록에서 구성 요소 이동

객체 이름과 해당 객체를 배치할 위치를 메서드의 매개 변수로 지정하여 addChildAt() 메서드를 호출하면 표시 목록에서의 객체 위치와 표시 심도를 변경할 수 있습니다. 예를 들어, 앞의 예제에 다음 코드를 추가하면 NumericStepper를 맨 위에 배치하고 루프를 반복하여 표시 목록에서 구성 요소의 새 위치를 표시할 수 있습니다.

```
this.addChildAt(aNs, numChildren - 1);
i = 0;
while(i < numChildren) {
    trace(getChildAt(i).name + " is at position: " + i++);
}
```

[출력] 패널에 다음과 같이 표시됩니다.

```
aNs is at position: 0
aButton is at position: 1
aCb is at position: 2
aButton is at position: 0
aCb is at position: 1
aNs is at position: 2
```

또한 NumericStepper는 화면에서 다른 구성 요소 앞에 표시되어야 합니다.

numChildren은 표시 목록에 있는 객체 수(1~**n**)이고 목록에서의 첫 번째 위치는 0입니다. 따라서 목록에 객체가 세 개 있는 경우 세 번째 객체의 인덱스 위치는 2입니다. 이 경우 표시 목록에서의 마지막 위치(또는 표시 심도를 고려했을 때 맨 위에 있는 객체)는 numChildren - 1이라는 것을 알 수 있습니다.

표시 목록에서 구성 요소 제거

removeChild() 및 removeChildAt() 메서드를 사용하면 표시 객체 컨테이너와 해당 표시 목록에서 구성 요소를 제거할 수 있습니다. 다음 예제에서는 스테이지에 세 개의 Button 구성 요소를 서로 겹치게 배치하고 각 구성 요소에 대한 이벤트 리스너를 추가합니다. 각 Button을 클릭하면 이벤트 핸들러가 해당 구성 요소를 표시 목록과 스테이지에서 제거합니다.

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널의 Button을 [라이브러리] 패널로 드래그합니다.
3. [액션] 패널을 열고 기본 타임라인에서 [프레임 1]을 선택한 후 다음 코드를 추가합니다.

```
import fl.controls.Button;

var i:int = 0;
while(i++ < 3) {
    makeButton(i);
}
function removeButton(event:MouseEvent):void {
    removeChildAt(numChildren -1);
}
function makeButton(num) {
    var aButton:Button = new Button();
    aButton.name = "Button" + num;
    aButton.label = aButton.name;
    aButton.move(200, 200);
    addChild(aButton);
    aButton.addEventListener(MouseEvent.CLICK, removeButton);
}
```

표시 목록에 대한 자세한 내용은 **ActionScript 3.0** 프로그래밍의 "디스플레이 프로그래밍"을 참조하십시오.

FocusManager를 사용한 작업

사용자가 Tab 키를 눌러 Flash 응용 프로그램에서 이동하거나 응용 프로그램에서 마우스를 클릭하면 FocusManager 클래스는 입력 포커스를 받을 구성 요소를 결정합니다. 구성 요소를 직접 만드는 경우를 제외하고는 FocusManager 인스턴스를 응용 프로그램에 직접 추가하거나 FocusManager를 활성화하는 코드를 작성할 필요가 없습니다.

RadioButton 객체가 포커스를 받으면 FocusManager는 해당 객체 및 groupName 값이 동일한 모든 객체를 검사한 후 selected 속성이 true로 설정된 객체에 포커스를 설정합니다.

각 모달 Window 구성 요소에 FocusManager의 인스턴스가 포함되어 있으므로 해당 윈도우 내에서만 자체적으로 탭이 설정됩니다. 따라서 사용자가 실수로 Tab 키를 눌러도 다른 윈도우의 구성 요소로 이동하지 않습니다.

FocusManager는 기본 탐색 체계 또는 **탭 루프**로 컨테이너의 요소 심도 레벨(또는 **z** 순서)을 사용합니다. 탭 루프는 대개 Tab 키를 사용하여 탐색합니다. 포커스가 있는 첫 번째 구성 요소에서 마지막 구성 요소로 포커스가 이동한 후 다시 첫 번째 구성 요소로 이동합니다. 심도 레벨은 주로 구성 요소를 스테이지로 드래그하는 순서에 따라 설정되지만 [수정] > [정렬] > [맨 앞으로 가져오기]/[맨 뒤로 보내기] 명령을 사용하여 최종 **z** 순서를 결정할 수도 있습니다. 심도 레벨에 대한 자세한 내용은 22페이지의 "표시 목록을 사용한 작업"을 참조하십시오.

응용 프로그램에서 구성 요소 인스턴스에 포커스를 설정하려면 setFocus() 메서드를 호출합니다. 예를 들어, 다음 예제에서는 현재 컨테이너(this)에 대해 FocusManager 인스턴스를 만들고 Button 인스턴스인 aButton에 포커스를 설정합니다.

```
var fm:FocusManager = new FocusManager(this);
fm.setFocus(aButton);
```

getFocus() 메서드를 호출하여 포커스가 있는 구성 요소를 확인하고 getNextFocusManagerComponent() 메서드를 호출하여 탭 루프에서 포커스를 받을 다음 구성 요소를 확인할 수 있습니다. 다음 예제에서는 CheckBox, RadioButton 및 Button이 스테이지에 있고 MouseEvent.CLICK 및 FocusEvent.MOUSE_FOCUS_CHANGE 이벤트에 대한 리스너가 각 구성 요소에 있습니다. MouseEvent.CLICK 이벤트가 발생하면 사용자가 구성 요소를 클릭한 것이므로 showFocus() 함수는 getNextFocusManagerComponent() 메서드를 호출하여 탭 루프에서 포커스를 받을 다음 구성 요소를 확인합니다. 그런 다음 setFocus() 메서드를 호출하여 해당 구성 요소에 포커스를 설정합니다. FocusEvent.MOUSE_FOCUS_CHANGE 이벤트가 발생하면 fc() 함수는 이 이벤트가 발생한 구성 요소의 이름을 표시합니다. 이 이벤트는 탭 루프의 다음 구성 요소가 아닌 해당 구성 요소를 클릭할 때 트리거됩니다.

```
// This example assumes a CheckBox (aCh), a RadioButton (aRb) and a Button
// (aButton) have been placed on the Stage.

import fl.managers.FocusManager;
import flash.display.InteractiveObject;

var fm:FocusManager = new FocusManager(this);

aCh.addEventListener(MouseEvent.CLICK, showFocus);
aRb.addEventListener(MouseEvent.CLICK, showFocus);
aButton.addEventListener(MouseEvent.CLICK, showFocus);
aCh.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
aRb.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
aButton.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);

function showFocus(event:MouseEvent):void {
    var nextComponent:InteractiveObject = fm.getNextFocusManagerComponent();
    trace("Next component in tab loop is: " + nextComponent.name);
    fm.setFocus(nextComponent);
}

function fc(fe:FocusEvent):void {
    trace("Focus Change: " + fe.target.name);
}
```

사용자가 Enter 키(Windows) 또는 Return 키(Macintosh)를 누를 때 포커스를 받는 Button을 만들려면 다음 코드와 같이 기본 Button으로 사용할 Button 인스턴스로 FocusManager.defaultButton 속성을 설정합니다.

```
import fl.managers.FocusManager;

var fm:FocusManager = new FocusManager(this);
fm.defaultButton = okButton;
```

FocusManager 클래스는 기본 Flash Player 포커스 사각형을 무시하고 모서리가 둥근 사용자 정의 포커스 사각형을 그립니다.

Flash 응용 프로그램에서 포커스 체계 만들기에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**에서 FocusManager 클래스를 참조하십시오. 사용자 정의 포커스 관리자를 만들려면 **IFocusManager** 인터페이스를 구현하는 클래스를 만들어야 합니다. 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**에서 IFocusManager를 참조하십시오.

목록 기반 구성 요소를 사용한 작업

List, DataGrid 및 TileList 구성 요소는 모두 SelectableList 기본 클래스에서 상속되기 때문에 이들 구성 요소는 목록 기반 구성 요소로 간주됩니다. 텍스트 상자와 목록으로 구성되는 ComboBox도 마찬가지로 목록 기반 구성 요소입니다.

List는 행으로 이루어져 있습니다. DataGrid와 TileList는 여러 열로 나뉠 수 있는 행으로 구성됩니다. 행과 열이 교차하는 위치를 셀이라고 합니다. 여러 행이 단일 열로 구성된 List의 경우에는 각 행이 셀입니다. 셀은 다음과 같은 두 가지 측면에서 매우 중요합니다.

- 셀에 들어 있는 데이터 값을 항목이라고 합니다. **항목**은 List에 정보 단위를 저장하는 데 사용되는 ActionScript 객체입니다. List를 배열로 간주하면 항목은 인덱싱된 각 배열 공간에 해당합니다. 일반적으로 List에서 항목은 표시되는 label 속성과 데이터를 저장하는 데 사용되는 data 속성을 갖는 객체입니다. **데이터 공급자**는 List의 항목에 대한 데이터 모델입니다. 데이터 공급자를 구성 요소의 dataProvider 속성에 지정하면 목록 기반 구성 요소를 간단히 채울 수 있습니다.
- 셀에는 텍스트부터 이미지, 무비 클립 또는 만드는 모든 클래스에 이르기까지 다양한 종류의 데이터를 저장할 수 있습니다. 이러한 이유로 셀은 해당 내용에 적합한 방식으로 그리거나 렌더링해야 합니다. 따라서 목록 기반 구성 요소에는 셀을 렌더링하는 데 사용되는 **셀 렌더러**가 있습니다. DataGrid의 경우 각 열은 DataGridColumn 객체이며, 내용에 맞게 각 열을 렌더링할 수 있도록 이 객체에도 cellRenderer 속성이 있습니다.

  모든 목록 기반 구성 요소에는 각 구성 요소의 셀을 로드하고 렌더링하기 위해 설정할 수 있는 cellRenderer와 dataProvider 속성이 있습니다. 이러한 속성 및 목록 기반 구성 요소를 사용한 작업에 대한 자세한 내용은 26페이지의 “DataProvider를 사용한 작업” 및 33페이지의 “CellRenderer를 사용한 작업”을 참조하십시오.

DataProvider를 사용한 작업

DataProvider는 ComboBox, DataGrid, List 및 TileList 구성 요소에 데이터를 제공하는 데 사용할 수 있는 데이터 소스입니다. 이들 구성 요소 클래스에는 dataProvider 속성이 하나씩 있으며, 이 속성에는 해당 구성 요소의 셀을 데이터로 채우는 데 사용할 수 있는 DataProvider 객체를 할당할 수 있습니다. 일반적으로 데이터 공급자는 Array 객체나 XML 객체와 같은 데이터 컬렉션입니다.

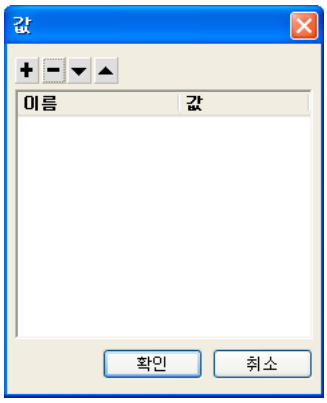
DataProvider 만들기

ComboBox, List 및 TileList 구성 요소의 경우에는 제작 환경에서 dataProvider 매개 변수를 사용하여 DataProvider를 만들 수 있습니다. 여러 열을 포함할 수 있는 DataGrid 구성 요소의 경우에는 데이터 공급자가 더 복잡하기 때문에 속성 관리자에 dataProvider 매개 변수가 없습니다. ActionScript를 사용하여 DataGrid를 비롯한 이들 구성 요소의 DataProvider를 만들 수도 있습니다.

dataProvider 매개 변수 사용

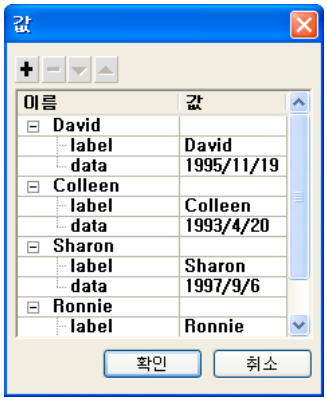
속성 관리자나 [구성 요소 관리자]의 [매개 변수] 탭에서 dataProvider 매개 변수를 클릭하여 ComboBox, List 및 TileList 구성 요소의 데이터 공급자를 간단히 만들 수 있습니다.

기본적으로 빈 Array가 표시되어 있는 값 셀을 두 번 클릭하면 [값] 대화 상자가 열립니다. 이 대화 상자에서 여러 레이블과 데이터 값을 입력하여 데이터 공급자를 만들 수 있습니다.



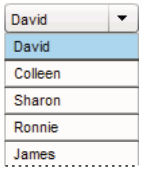
dataProvider의 값 대화 상자

dataProvider에 항목을 추가하려면 더하기 기호를 클릭하고 항목을 삭제하려면 빼기 기호를 클릭합니다. 위쪽 화살표나 아래쪽 화살표를 클릭하면 선택한 항목을 목록에서 위로 또는 아래로 이동할 수 있습니다. 다음 그림에서는 자녀의 이름과 생일 목록을 만드는 [값] 대화 상자를 보여 줍니다.



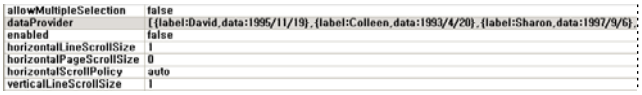
데이터가 입력된 값 대화 상자

여기서 만든 Array는 레이블과 값 필드 쌍으로 구성됩니다. 레이블 필드는 label과 data이며 값 필드는 각 자녀의 이름과 생일입니다. 레이블 필드는 List에 표시되는 내용, 즉 이 경우에는 자녀의 이름을 나타냅니다. 이렇게 만든 ComboBox의 모양은 다음과 같습니다.



DataProvider로 채워진 ComboBox

데이터를 모두 추가한 후에는 [확인]을 클릭하여 대화 상자를 닫습니다. 이제 dataProvider 매개 변수의 Array가 방금 만든 항목으로 채워집니다.



데이터가 있는 dataProvider 매개 변수

ActionScript를 통해 구성 요소의 dataProvider 속성에 액세스하여 레이블 및 데이터 값에 액세스할 수도 있습니다.

ActionScript를 사용하여 DataProvider 만들기

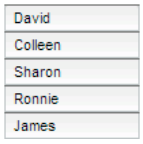
Array 또는 XML 객체에 데이터를 만든 다음 이 객체를 DataProvider 생성자에 value 매개 변수로 제공하여 DataProvider를 만들 수 있습니다.

참고: ActionScript 3.0에서는 dataProvider 속성이 DataProvider 객체로 정의되어 있기 때문에 이 속성에는 DataProvider 형식의 객체만 할당할 수 있고 Array 또는 XML 객체는 직접 할당할 수 없습니다.

다음 예제에서는 행이 여러 개이고 열이 하나인 List 구성 요소를 여러 명의 자녀 이름과 생일로 채웁니다. 이 예제에서는 items Array에 목록을 정의한 다음, DataProvider 인스턴스(new DataProvider(items))를 만들고 List 구성 요소의 dataProvider 속성에 이 인스턴스를 할당할 때 이 목록을 매개 변수로 제공합니다.

```
import fl.controls.List;
import fl.data.DataProvider;

var aList:List = new List();
var items:Array = [
{label:"David", data:"11/19/1995"},
{label:"Colleen", data:"4/20/1993"},
{label:"Sharon", data:"9/06/1997"},
{label:"Ronnie", data:"7/6/1993"},
{label:"James", data:"2/15/1994"},
];
aList.dataProvider = new DataProvider(items);
addChild(aList);
aList.move(150,150);
```

Array는 레이블과 값 필드 쌍으로 구성됩니다. 레이블 필드는 label과 data이며 값 필드는 각 자녀의 이름과 생일입니다. 레이블 필드는 List에 표시되는 내용, 즉 이 경우에는 자녀의 이름을 나타냅니다. 이렇게 만든 List의 모양은 다음과 같습니다.

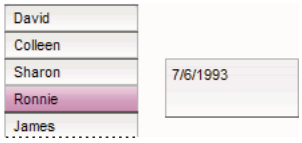


DataProvider로 채워진 List

사용자가 목록에서 항목을 클릭하여 선택하면 change 이벤트가 발생하고, 그 결과로 데이터 필드의 값을 사용할 수 있습니다. 다음 예제에서는 사용자가 목록에서 이름을 선택할 때 자녀의 생일이 표시되도록 이전 예제에 TextArea(aTa)와 이벤트 핸들러(changeHandler)를 추가합니다.

```
import fl.controls.List;
import fl.controls.TextArea;
import flash.events.Event;
import fl.data.DataProvider;

var aList:List = new List();
var aTa:TextArea = new TextArea();
var items:Array = [
{label:"David", data:"1/19/1995"},
{label:"Colleen", data:"4/20/1993"},
{label:"Sharon", data:"9/06/1994"},
{label:"Ronnie", data:"7/6/1993"},
{label:"James", data:"2/15/1994"},
];
aList.dataProvider = new DataProvider(items);

addChild(aList);
addChild(aTa);

aList.move(150,150);
aTa.move(150, 260);

aList.addEventListener(Event.CHANGE, changeHandler);

function changeHandler(event:Event):void {
    aTa.text = event.target.selectedItem.data;
};
```

이제 사용자가 List에서 자녀의 이름을 선택하면 다음 그림과 같이 해당 자녀의 생일이 TextArea에 표시됩니다. 이를 위해 changeHandler() 함수는 TextArea(aTa.text)의 text 속성을 선택된 항목(event.target.selectedItem.data)의 데이터 필드 값으로 설정합니다. event.target 속성은 이벤트를 트리거한 객체, 즉 List입니다.



List에 대한 DataProvider의 데이터 필드 표시

DataProvider에는 텍스트 이외의 데이터를 포함할 수 있습니다. 다음 예제에서는 TileList에 데이터를 제공하는 DataProvider에 무비 클립을 포함합니다. 이 예제에서는 색상이 설정된 무비 클립을 만든 다음 addItem() 호출을 통해 각 항목을 추가하여 DataProvider를 만듭니다.

```
import fl.data.DataProvider;
import flash.display.DisplayObject;

var aBox:MovieClip = new MovieClip();
var i:uint = 0;
var colors:Array = new Array(0x00000, 0xFF0000, 0x0000CC, 0x00CC00, 0xFFFF00);
var colorNames:Array = new Array("Midnight", "Cranberry", "Sky", "Forest", "July");
var dp:DataProvider = new DataProvider();
for(i=0; i < colors.length; i++) {
    drawBox(aBox, colors[i]);// draw box w next color in array
    dp.addItem( {label:colorNames[i], source:aBox} );
}
aTl.dataProvider = dp;
aTl.columnWidth = 110;
aTl.rowHeight = 130;
aTl.setSize(280,150);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(0, 0, 100, 100);
            box.graphics.endFill();
```

배열 대신 XML 데이터를 사용하여 DataProvider 객체를 채울 수도 있습니다. 예를 들어, 다음 코드는 employeesXML이라는 XML 객체에 데이터를 저장한 후 이 객체를 DataProvider() 생성자 함수의 값 매개 변수로 전달합니다.

```
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aDg:DataGrid = new DataGrid();
addChild(aDg);

var employeesXML:XML =
    <employees>
        <employee Name="Edna" ID="22" />
        <employee Name="Stu" ID="23" />
    </employees>;

var myDP:DataProvider = new DataProvider(employeesXML);

aDg.columns = ["Name", "ID"];
aDg.dataProvider = myDP;
```

데이터를 앞의 코드와 같이 XML 데이터의 특성으로 제공하거나 다음 코드와 같이 XML 데이터의 속성으로 제공할 수 있습니다.

```
var employeesXML:XML =
    <employees>
        <employee>
            <Name>Edna</Name>
            <ID>22</ID>
        </employee>
        <employee>
            <Name>Stu</Name>
            <ID>23</ID>
        </employee>
    </employees>;
```

DataProvider에는 DataProvider를 액세스하고 조작하는 데 사용할 수 있는 여러 가지 메서드와 속성이 있습니다. DataProvider API를 사용하여 DataProvider에서 항목을 추가, 제거, 바꾸기, 정렬 및 병합할 수 있습니다.

DataProvider 조작

addItem() 및 addItemAt() 메서드는 DataProvider에 항목을 추가하는 데 사용됩니다. 다음 예제에서는 사용자가 편집 가능한 ComboBox의 텍스트 필드에 입력하는 항목을 추가합니다. 이 예제에서는 ComboBox를 스테이지에 드래그하고 인스턴스 이름을 aCb로 지정했다고 가정합니다.

```
import fl.data.DataProvider;
import fl.events.ComponentEvent;

var items:Array = [
{label:"Roger"},
{label:"Carolyn"},
{label:"Darrell"},
{label:"Rebecca"},
{label:"Natalie"},
{label:"Mitchell"},
];
aCb.dataProvider = new DataProvider(items);

aCb.addEventListener(ComponentEvent.ENTER, newItemHandler);

function newItemHandler(event:ComponentEvent):void {
    var newRow:int = event.target.length + 1;
        event.target.addItemAt({label:event.target.selectedLabel},
        event.target.length);
}
```

DataProvider를 통해 구성 요소의 항목을 바꾸거나 제거할 수도 있습니다. 다음 예제에서는 listA와 listB라는 두 개의 List 구성 요소를 구현하고 Sync라는 레이블이 지정된 Button을 제공합니다. 사용자가 Button을 클릭하면 replaceItemAt() 메서드를 통해 listB의 항목이 listA의 항목으로 바뀝니다. listA가 listB보다 긴 경우에는 addItem() 메서드를 호출하여 listB에 항목을 더 추가합니다. 또한 listB가 listA보다 긴 경우에는 removeItemAt() 메서드를 호출하여 ListB의 남는 항목을 제거합니다.

```
// Requires the List and Button components to be in the library

import fl.controls.List;
import fl.controls.Button;
import flash.events.Event;
import fl.data.DataProvider;

var listA:List = new List();
var listB:List = new List();
var syncButton:Button = new Button();
syncButton.label = "Sync";

var itemsA:Array = [
{label:"David"},
{label:"Colleen"},
{label:"Sharon"},
{label:"Ronnie"},
{label:"James"},
];
var itemsB:Array = [
{label:"Roger"},
{label:"Carolyn"},
{label:"Darrell"},
{label:"Rebecca"},
{label:"Natalie"},
{label:"Mitchell"},
];
listA.dataProvider = new DataProvider(itemsA);
listB.dataProvider = new DataProvider(itemsB);
```

```
addChild(listA);
addChild(listB);
addChild(syncButton);

listA.move(100, 100);
listB.move(250, 100);
syncButton.move(175, 220);

syncButton.addEventListener(MouseEvent.CLICK, syncHandler);

function syncHandler(event:MouseEvent):void {
    var i:uint = 0;
    if(listA.length > listB.length) { //if listA is longer, add items to B
        while(i < listB.length) {
            listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i), i);
            ++i;
        }
        while(i < listA.length) {
            listB.dataProvider.addItem(listA.dataProvider.getItemAt(i++));
        }
    } else if(listA.length == listB.length) { //if listA and listB are equal length
        while(i < listB.length) {
            listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i), i);
            ++i;
        }
    } else {         //if listB is longer, remove extra items from B
        while(i < listA.length) {
            listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i), i);
            ++i;
        }
        while(i < listB.length) {
            listB.dataProvider.removeItemAt(i++);
        }
    }
}
```

merge(), sort() 및 sortOn() 메서드를 사용하여 DataProvider를 병합하거나 정렬할 수도 있습니다. 다음은 aDg와 bDg라는 두 개의 DataGrid 인스턴스를 소프트볼 팀 명단으로 채웁니다. 그런 다음 레이블이 Merge인 Button을 추가합니다. 사용자가 이 Button을 클릭하면 이벤트 핸들러(mrgHandler)가 bDg의 명단과 aDg의 명단을 병합하고 Name 열에서 결과 DataGrid를 정렬합니다.

```
import fl.data.DataProvider;
import fl.controls.DataGrid;
import fl.controls.Button;

var aDg:DataGrid = new DataGrid();
var bDg:DataGrid = new DataGrid();
var mrgButton:Button = new Button();
addChild(aDg);
addChild(bDg);
addChild(mrgButton);
bldRosterGrid(aDg);
bldRosterGrid(bDg);
var aRoster:Array = new Array();
var bRoster:Array = new Array();
aRoster = [
        {Name:"Wilma Carter", Bats:"R", Throws:"R", Year:"So", Home: "Redlands, CA"},
        {Name:"Sue Pennypacker", Bats:"L", Throws:"R", Year:"Fr", Home: "Athens, GA"},
        {Name:"Jill Smithfield", Bats:"R", Throws:"L", Year:"Sr", Home: "Spokane, WA"},
        {Name:"Shirley Goth", Bats:"R", Throws:"R", Year:"Sr", Home: "Carson, NV"}
];
bRoster = [
```

```
        {Name:"Angelina Davis", Bats:"R", Throws:"R", Year:"So", Home: "Odessa, TX"},
        {Name:"Maria Santiago", Bats:"L", Throws:"L", Year:"Sr", Home: "Tacoma, WA"},
        {Name:"Debbie Ferguson", Bats:"R", Throws:"R", Year: "Jr", Home: "Bend, OR"}
];
aDg.dataProvider = new DataProvider(aRoster);
bDg.dataProvider = new DataProvider(bRoster);
aDg.move(50,50);
aDg.rowCount = aDg.length;
bDg.move(50,200);
bDg.rowCount = bDg.length;
mrgButton.label = "Merge";
mrgButton.move(200, 315);
mrgButton.addEventListener(MouseEvent.CLICK, mrgHandler);

function bldRosterGrid(dg:DataGrid){
    dg.setSize(400, 300);
    dg.columns = ["Name", "Bats", "Throws", "Year", "Home"];
    dg.columns[0].width = 120;
    dg.columns[1].width = 50;
    dg.columns[2].width = 50;
    dg.columns[3].width = 40;
    dg.columns[4].width = 120;
};

function mrgHandler(event:MouseEvent):void {
    aDg.dataProvider.merge(bDg.dataProvider);
    aDg.dataProvider.sortOn("Name");
}
```

자세한 내용은 ActionScript 3.0 참조 설명서를 참조하십시오.

CellRenderer를 사용한 작업

CellRenderer는 List, DataGrid, TileList 및 ComboBox 같은 목록 기반 구성 요소에서 행의 사용자 정의 셀 내용을 조작하고 표시하는 데 사용할 수 있는 클래스입니다. 사용자 정의된 셀에는 텍스트, 미리 만들어진 구성 요소(예: CheckBox) 또는 사용자가 만드는 표시 객체 클래스를 포함할 수 있습니다. 사용자 정의 CellRenderer를 사용하여 데이터를 렌더링하기 위해 CellRenderer 클래스를 확장하거나 ICellRenderer 인터페이스를 구현하여 사용자 정의 CellRenderer 클래스를 직접 만들 수 있습니다.

List, DataGrid, TileList 및 ComboBox 클래스는 SelectableList 클래스의 하위 클래스입니다. SelectableList 클래스에는 cellRenderer 스타일이 들어 있는데 이 스타일은 구성 요소에서 셀을 렌더링하는 데 사용하는 표시 객체를 정의합니다.

List 객체의 setRendererStyle() 메서드를 호출하면 CellRenderer에서 사용하는 스타일의 서식을 조정할 수 있습니다(33페이지의 “셀 서식 지정” 참조). 또는 CellRenderer로 사용할 사용자 정의 클래스를 직접 정의할 수 있습니다(34페이지의 “사용자 정의 CellRenderer 클래스 정의” 참조).

셀 서식 지정

CellRenderer 클래스에는 셀의 서식을 제어하는 데 사용할 수 있는 여러 가지 스타일이 포함되어 있습니다.

다음 스타일을 사용하면 셀의 여러 가지 상태(비활성, 다운, 오버 및 업)에 사용되는 스킨을 정의할 수 있습니다.

- disabledSkin 및 selectedDisabledSkin
- downSkin 및 selectedDownSkin
- overSkin 및 selectedOverSkin
- upSkin 및 selectedUpSkin

다음은 텍스트 서식에 적용되는 스타일입니다.

- disabledTextFormat
- textFormat
- textPadding

List 객체의 setRendererStyle() 메서드나 CellRenderer 객체의 setStyle() 메서드를 호출하여 이러한 스타일을 설정할 수 있으며, List 객체의 getRendererStyle() 메서드나 CellRenderer 객체의 getStyle() 메서드를 호출하여 이러한 스타일을 가져올 수 있습니다. List 객체의 rendererStyles 속성이나 CellRenderer 객체의 getStyleDefinition() 메서드를 통해 모든 렌더러 스타일을 객체의 명명된 속성으로 정의하는 객체에 액세스할 수도 있습니다.

clearRendererStyle() 메서드를 호출하면 스타일을 기본값으로 재설정할 수 있습니다.

목록의 행 높이를 가져오거나 설정하려면 List 객체의 rowHeight 속성을 사용합니다.

사용자 정의 CellRenderer 클래스 정의

CellRenderer 클래스를 확장하여 사용자 정의 CellRenderer를 정의하는 클래스 만들기

예를 들어, 다음 코드에서는 두 가지 클래스를 사용합니다. ListSample 클래스는 List 구성 요소를 인스턴스화하고 다른 클래스인 CustomRenderer를 사용하여 List 구성 요소에 사용할 셀 렌더러를 정의합니다. CustomRenderer 클래스는 CellRenderer 클래스를 확장합니다.

1. [파일] > [새로 만들기]를 선택합니다.
2. 표시되는 [새 문서] 대화 상자에서 [Flash 파일(ActionScript 3.0)]을 선택하고 [확인]을 클릭합니다.
3. [윈도우] > [구성 요소]를 선택하여 [구성 요소] 패널을 표시합니다.
4. [구성 요소] 패널에서 List 구성 요소를 스테이지로 드래그합니다.
5. 속성 관리자가 보이지 않으면 [윈도우] > [속성] > [속성}을 선택합니다.
6. List 구성 요소를 선택하고 속성 관리자에서 다음 속성을 설정합니다.
   - 인스턴스 이름: myList
   - W(폭): 200
   - H(높이): 300
   - X: 20
   - Y: 20
7. 타임라인에서 레이어 1의 프레임 1을 선택하고 [윈도우] > [액션]을 선택합니다.
8. [액션] 패널에 다음 스크립트를 입력합니다.

```
myList.setStyle("cellRenderer", CustomCellRenderer);
myList.addItem({label:"Burger -- $5.95"});
myList.addItem({label:"Fries -- $1.95"});
```

9. [파일] > [저장]을 선택합니다. 파일에 이름을 지정한 다음 [확인] 버튼을 클릭합니다.
10. [파일] > [새로 만들기]를 선택합니다.
11. 표시되는 [새 문서] 대화 상자에서 [ActionScript 파일]을 선택하고 [확인] 버튼을 클릭합니다.
12. 스크립트 윈도우에 다음 코드를 입력하여 CustomCellRenderer 클래스를 정의합니다.

```
package {
    import fl.controls.listClasses.CellRenderer;
    import flash.text.TextFormat;
    import flash.filters.BevelFilter;
    public class CustomCellRenderer extends CellRenderer {
        public function CustomCellRenderer() {
            var format:TextFormat = new TextFormat("Verdana", 12);
            setStyle("textFormat", format);
            this.filters = [new BevelFilter()];
        }
    }
}
```

13 [파일] > [저장]을 선택합니다. 파일 이름을 CustomCellRenderer.as로 지정하고 저장 위치로 FLA 파일과 같은 디렉토리를 지정한 후 [확인] 버튼을 클릭합니다.

14 [컨트롤] > [무비 테스트]를 선택합니다.

ICellRenderer 인터페이스를 구현하는 클래스를 사용하여 사용자 정의 CellRenderer 정의

DisplayObject 클래스에서 상속되고 ICellRenderer 인터페이스를 구현하는 모든 클래스를 사용하여 CellRenderer를 정의할 수 있습니다. 예를 들어, 다음 코드에서는 두 가지 클래스를 정의합니다. ListSample2 클래스는 표시 목록에 List 객체를 추가하고 CustomRenderer 클래스를 사용하도록 이 객체의 CellRenderer를 정의합니다. CustomRenderer 클래스는 DisplayObject 클래스를 확장하는 CheckBox 클래스를 확장하고 ICellRenderer 인터페이스를 구현합니다. 이때 CustomRenderer 클래스는 ICellRenderer 인터페이스에 정의되어 있는 data 및 listData 속성에 대한 getter 및 setter 메서드를 정의합니다. ICellRenderer 인터페이스에 정의되어 있는 다른 속성과 메서드(selected 속성 및 setSize() 메서드)는 CheckBox 클래스에 이미 정의되어 있습니다.

1 [파일] > [새로 만들기]를 선택합니다.

2 표시되는 [새 문서] 대화 상자에서 [Flash 파일(ActionScript 3.0)]을 선택하고 [확인]을 클릭합니다.

3 [윈도우] > [구성 요소]를 선택하여 [구성 요소] 패널을 표시합니다.

4 [구성 요소] 패널에서 List 구성 요소를 스테이지로 드래그합니다.

5 속성 관리자가 보이지 않으면 [윈도우] > [속성] > [속성}을 선택합니다.

6 List 구성 요소를 선택하고 속성 관리자에서 다음 속성을 설정합니다.

- 인스턴스 이름: myList
- W(폭): 100
- H(높이): 300
- X: 20
- Y: 20

7 타임라인에서 레이어 1의 프레임 1을 선택하고 [윈도우] > [액션]을 선택합니다.

8 [액션] 패널에 다음 스크립트를 입력합니다.

```
myList.setStyle("cellRenderer", CustomCellRenderer);
myList.addItem({name:"Burger", price:"$5.95"});
myList.addItem({name:"Fries", price:"$1.95"});
```

9 [파일] > [저장]을 선택합니다. 파일에 이름을 지정한 다음 [확인] 버튼을 클릭합니다.

10 [파일] > [새로 만들기]를 선택합니다.

11 표시되는 [새 문서] 대화 상자에서 [ActionScript 파일]을 선택하고 [확인] 버튼을 클릭합니다.

12 스크립트 윈도우에 다음 코드를 입력하여 CustomCellRenderer 클래스를 정의합니다.

```
package
{
    import fl.controls.CheckBox;
    import fl.controls.listClasses.ICellRenderer;
    import fl.controls.listClasses.ListData;
    public class CustomCellRenderer extends CheckBox implements ICellRenderer {
        private var _listData:ListData;
        private var _data:Object;
        public function CustomCellRenderer() {
        }
        public function set data(d:Object):void {
            _data = d;
            label = d.label;
        }
        public function get data():Object {
            return _data;
        }
        public function set listData(ld:ListData):void {
            _listData = ld;
        }
        public function get listData():ListData {
            return _listData;
        }
    }
}
```

13 [파일] > [저장]을 선택합니다. 파일 이름을 CustomCellRenderer.as로 지정하고 저장 위치로 FLA 파일과 같은 디렉토리를 지정한 후 [확인] 버튼을 클릭합니다.

14 [컨트롤] > [무비 테스트]를 선택합니다.

심볼을 사용하여 CellRenderer 정의

라이브러리에 있는 심볼을 사용하여 CellRenderer를 정의할 수도 있습니다. 이 경우 ActionScript에 사용할 수 있도록 심볼을 내보내야 하며, 라이브러리 심볼의 클래스 이름에는 ICellRenderer 인터페이스를 구현하거나 CellRenderer 클래스 또는 그 하위 클래스 중 하나를 확장하는 클래스 파일이 연결되어 있어야 합니다.

다음 예제에서는 라이브러리 심볼을 사용하여 사용자 정의 CellRenderer를 정의합니다.

1 [파일] > [새로 만들기]를 선택합니다.

2 표시되는 [새 문서] 대화 상자에서 [Flash 파일(ActionScript 3.0)]을 선택하고 [확인]을 클릭합니다.

3 [윈도우] > [구성 요소]를 선택하여 [구성 요소] 패널을 표시합니다.

4 [구성 요소] 패널에서 List 구성 요소를 스테이지로 드래그합니다.

5 속성 관리자가 보이지 않으면 [윈도우] > [속성] > [속성}을 선택합니다.

6 List 구성 요소를 선택하고 속성 관리자에서 다음 속성을 설정합니다.

- 인스턴스 이름: myList
- W(폭): 100
- H(높이): 400
- X: 20
- Y: 20

7 [매개 변수] 패널을 클릭하고 dataProvider 행의 두 번째 열을 두 번 클릭합니다.

8 표시되는 [값] 대화 상자에서 더하기 기호를 두 번 클릭하여 레이블이 각각 label0과 label1로 설정된 두 개의 데이터 요소를 추가한 후 [확인] 버튼을 클릭합니다.

9 [텍스트 도구]를 사용하여 스테이지에 텍스트 필드를 그립니다.

10 텍스트 필드를 선택하고 속성 관리자에서 다음 속성을 설정합니다.

- 텍스트 유형: 동적 텍스트
- 인스턴스 이름: textField
- W(폭): 100
- 글꼴 크기: 24
- X: 0
- Y: 0

11 텍스트 필드를 선택한 상태에서 [수정] > [심볼로 변환]을 선택합니다.

12 [심볼로 변환] 대화 상자에서 다음과 같이 설정하고 [확인] 버튼을 클릭합니다.

- 이름: MyCellRenderer
- 유형: 무비 클립
- ActionScript에 내보내기: 선택
- 첫 프레임으로 내보내기: 선택
- 클래스: MyCellRenderer
- 기본 클래스: flash.display.MovieClip

  ActionScript 클래스 경고가 표시되면 경고 상자에 있는 [확인] 버튼을 클릭하십시오.

13 스테이지에서 새 무비 클립 심볼의 인스턴스를 삭제합니다.

14 타임라인에서 레이어 1의 프레임 1을 선택하고 [윈도우] > [액션]을 선택합니다.

15 [액션] 패널에 다음 스크립트를 입력합니다.

```
myList.setStyle("cellRenderer", MyCellRenderer);
```

16 [파일] > [저장]을 선택합니다. 파일에 이름을 지정한 다음 [확인] 버튼을 클릭합니다.

17 [파일] > [새로 만들기]를 선택합니다.

18 표시되는 [새 문서] 대화 상자에서 [ActionScript 파일]을 선택하고 [확인] 버튼을 클릭합니다.

19 스크립트 윈도우에 다음 코드를 입력하여 MyCellRenderer 클래스를 정의합니다.

```
package {
    import flash.display.MovieClip;
    import flash.filters.GlowFilter;
    import flash.text.TextField;
    import fl.controls.listClasses.ICellRenderer;
    import fl.controls.listClasses.ListData;
    import flash.utils.setInterval;
    public class MyCellRenderer extends MovieClip implements ICellRenderer {
        private var _listData:ListData;
        private var _data:Object;
        private var _selected:Boolean;
        private var glowFilter:GlowFilter;
        public function MyCellRenderer() {
            glowFilter = new GlowFilter(0xFFFF00);
            setInterval(toggleFilter, 200);
        }
        public function set data(d:Object):void {
            _data = d;
            textField.text = d.label;
        }
```

```
        public function get data():Object {
            return _data;
        }
        public function set listData(ld:ListData):void {
            _listData = ld;
        }
        public function get listData():ListData {
            return _listData;
        }
        public function set selected(s:Boolean):void {
            _selected = s;
        }
        public function get selected():Boolean {
            return _selected;
        }
        public function setSize(width:Number, height:Number):void {
        }
        public function setStyle(style:String, value:Object):void {
        }
        public function setMouseState(state:String):void{
        }
        private function toggleFilter():void {
            if (textField.filters.length == 0) {
                textField.filters = [glowFilter];
            } else {
                textField.filters = [];
            }
        }
    }
}
```

20 [파일] > [저장]을 선택합니다. 파일 이름을 MyCellRenderer.as로 지정하고 저장 위치로 FLA 파일과 같은 디렉토리를 지정한 후 [확인] 버튼을 클릭합니다.

21 [컨트롤] > [무비 테스트]를 선택합니다.

CellRenderer 속성

data 속성은 CellRenderer에 설정된 모든 속성이 들어 있는 객체입니다. 예를 들어, Checkbox 클래스를 확장하는 사용자 정의 CellRenderer를 정의하는 다음 클래스에서 data 속성의 setter 함수는 CheckBox 클래스에서 상속되는 label 속성에 data.label의 값을 전달합니다.

```
    public class CustomRenderer extends CheckBox implements ICellRenderer {
        private var _listData:ListData;
        private var _data:Object;
        public function CustomRenderer() {
        }
        public function set data(d:Object):void {
            _data = d;
            label = d.label;
        }
        public function get data():Object {
            return _data;
        }
        public function set listData(ld:ListData):void {
            _listData = ld;
        }
        public function get listData():ListData {
            return _listData;
        }
    }
}
```

selected 속성은 목록에서 셀이 선택되어 있는지 여부를 정의합니다.

DataGrid 객체의 열에 CellRenderer 적용

DataGrid 객체에는 열이 여러 개 포함될 수 있으며 각 열에 서로 다른 셀 렌더러를 지정할 수 있습니다. DataGrid의 각 열은 DataGridColumn 객체로 표현되며 DataGridColumn 클래스에는 열의 CellRenderer를 정의할 수 있는 cellRenderer 속성이 포함됩니다.

편집 가능한 셀의 CellRenderer 정의

DataGridCellEditor 클래스는 DataGrid 객체의 편집 가능한 셀에 사용되는 렌더러를 정의합니다. 이 클래스는 DataGrid 객체의 editable 속성이 true로 설정된 경우에 사용자가 셀을 편집하기 위해 클릭했을 때 셀의 렌더러로 사용됩니다. 편집 가능한 셀의 CellRenderer를 정의하려면 DataGrid 객체의 columns 배열에 포함된 각 요소에 itemEditor 속성을 설정해야 합니다.

이미지, SWF 파일 또는 무비 클립을 CellRenderer로 사용

CellRenderer의 하위 클래스인 ImageCell 클래스는 셀의 주요 내용이 이미지, SWF 파일 또는 무비 클립인 셀을 렌더링하는 데 사용되는 객체를 정의합니다. ImageCell 클래스에는 셀 모양을 정의하는 데 사용할 수 있는 다음과 같은 스타일이 포함되어 있습니다.

- imagePadding — 셀 가장자리와 이미지 가장자리 사이의 패딩(픽셀)입니다.
- selectedSkin — 선택된 상태를 나타내는 데 사용되는 스킨입니다.
- textOverlayAlpha — 셀 레이블 뒤에 있는 오버레이의 불투명도입니다.
- textPadding — 셀 가장자리와 텍스트 가장자리 사이의 패딩(픽셀)입니다.

ImageCell 클래스는 TileList 클래스의 기본 CellRenderer입니다.

구성 요소를 액세스 가능하게 만들기

화면 내용을 읽어 주는 화면 판독기를 통해 시각 장애가 있는 사용자도 Flash 응용 프로그램의 시각적인 정보에 쉽게 액세스할 수 있습니다. Flash 응용 프로그램에서 화면 판독기에 액세스할 수 있게 만드는 방법에 대한 자세한 내용은 Flash 사용 안내서의 18장, 액세스 가능한 내용 만들기를 참조하십시오.

ActionScript 3.0 구성 요소를 화면 판독기를 통해 액세스할 수 있도록 하려면 화면 판독기의 액세스 가능성 클래스도 가져와서 이 클래스의 enableAccessibility() 메서드를 호출해야 합니다. 화면 판독기를 통해 액세스 가능하도록 만들 수 있는 ActionScript 3.0 구성 요소는 다음과 같습니다.

구성 요소	**Accessibility 클래스**
버튼	ButtonAccImpl
CheckBox	CheckBoxAccImpl
ComboBox	ComboBoxAccImpl
List	ListAccImpl
RadioButton	RadioButtonAccImpl
TileList	TileListAccImpl

구성 요소 액세스 가능성 클래스는 fl.accessibility 패키지에 들어 있습니다. 예를 들어, CheckBox 구성 요소를 화면 판독기를 통해 액세스할 수 있도록 만들려면 응용 프로그램에 다음 명령문을 추가해야 합니다.

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

구성 요소에 대한 액세스 가능성은 만드는 인스턴스 수에 관계없이 한 번만 활성화하면 됩니다.

참고: 액세스 가능성 기능을 부분적으로 설정하면 필요한 클래스가 컴파일하는 동안 추가되기 때문에 파일 크기가 커질 수 있습니다.

대부분의 구성 요소는 키보드를 사용하여 탐색할 수 있습니다. 액세스 가능한 구성 요소 활성화 및 키보드를 사용한 탐색에 대한 자세한 내용은 41페이지의 “UI 구성 요소 사용”의 사용자 상호 작용 단원과 **ActionScript 3.0 Reference for Flash Professional**의 accessibility 클래스를 참조하십시오.

4장: UI 구성 요소 사용

이 장에서는 Flash에 포함된 다음과 같은 ActionScript 3.0 UI(사용자 인터페이스) 구성 요소를 사용하는 방법에 대해 설명합니다.

Button 구성 요소 사용

버튼 구성 요소는 크기를 조절할 수 있는 사각형 버튼으로, 사용자가 마우스나 스페이스바로 눌러 응용 프로그램에서 액션을 시작할 수 있습니다. Button에는 사용자 정의 아이콘을 추가할 수 있습니다. Button의 비헤이비어를 누름에서 전환으로 변경할 수도 있습니다. 전환 Button은 클릭하면 눌린 상태로 있다가 다시 클릭하면 원래 상태로 되돌아옵니다.

Button은 많은 양식 및 웹 응용 프로그램에서 기본이 되는 요소입니다. 사용자가 이벤트를 시작하도록 하려는 경우 버튼을 사용할 수 있습니다. 예를 들어, 대부분의 양식에는 [전송] 버튼이 있고, 프리젠테이션에는 [이전] 및 [다음] 버튼을 추가할 수 있습니다.

Button 구성 요소와 사용자의 상호 작용

응용 프로그램에서 버튼을 활성화하거나 비활성화할 수 있습니다. 버튼이 비활성화된 상태에서는 마우스 또는 키보드 입력을 수신하지 못합니다. 활성화된 버튼은 클릭하거나 Tab 키를 눌러 이동하면 포커스를 받게 됩니다. Button 인스턴스가 포커스를 받으면 다음 키를 사용하여 제어할 수 있습니다.

키	설명
Shift+Tab	이전 객체로 포커스를 이동합니다.
스페이스바	버튼을 누르거나 놓고 click 이벤트를 트리거합니다.
탭	다음 객체로 포커스를 이동합니다.
Enter/Return	버튼이 FocusManager의 기본 Button으로 설정되어 있는 경우 다음 객체로 포커스를 이동합니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**의 IFocusManager 인터페이스 및 FocusManager 클래스와 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

각 Button 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다.

참고: 버튼보다 큰 아이콘은 버튼의 테두리를 넘어갑니다.

버튼을 응용 프로그램의 기본 누름 버튼(사용자가 Enter 키를 누를 때 click 이벤트를 받는 버튼)으로 지정하려면 FocusManager.defaultButton을 설정합니다. 예를 들어, 다음 코드에서는 submitButton이라는 Button 인스턴스를 기본 버튼으로 설정합니다.

```
FocusManager.defaultButton = submitButton;
```

응용 프로그램에 Button 구성 요소를 추가할 때 다음 ActionScript 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.ButtonAccImpl;

ButtonAccImpl.enableAccessibility();
```

구성 요소에 대한 액세스 가능성은 만드는 인스턴스 수에 관계없이 한 번만 활성화하면 됩니다.

Button 구성 요소 매개 변수

속성 관리자([윈도우] > [속성] > [속성])나 [구성 요소 관리자]([윈도우] > [구성 요소 관리자])에서 각 Button 인스턴스에 대해 emphasized, label, labelPlacement, selected, toggle 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수에 값을 지정하면 그에 따라 응용 프로그램에서 해당 속성의 초기 상태가 설정됩니다. 하지만 ActionScript에서 속성을 설정하면 매개 변수에 지정한 값이 재정의됩니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 Button 클래스를 참조하십시오.

Button 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램을 제작하는 동안 Button 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서는 Button을 클릭하면 ColorPicker 구성 요소의 상태가 변경됩니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널의 Button 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 입력합니다.
   - 인스턴스 이름으로 **aButton**을 입력합니다.
   - label 매개 변수로 **Show**를 입력합니다.
3. ColorPicker를 스테이지에 추가하고 인스턴스 이름을 **aCp**로 지정합니다.
4. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
aCp.visible = false;

aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {

    switch(event.currentTarget.label) {
        case "Show":
            aCp.visible = true;
            aButton.label = "Disable";
            break;
        case "Disable":
            aCp.enabled = false;
            aButton.label = "Enable";
            break;
        case "Enable":
            aCp.enabled = true;
            aButton.label = "Hide";
            break;
        case "Hide":
            aCp.visible = false;
            aButton.label = "Show";
            break;
    }
}
```

   두 번째 코드 행은 clickHandler() 함수를 MouseEvent.CLICK 이벤트의 이벤트 핸들러 함수로 등록합니다. 사용자가 Button을 클릭하면 이벤트가 발생하고 clickHandler() 함수가 Button 값에 따라 다음 액션 중 하나를 수행합니다.

   - Show는 ColorPicker를 보이게 하고 Button 레이블을 Disable로 변경합니다.
   - Disable은 ColorPicker를 비활성화하고 Button 레이블을 Enable로 변경합니다.
   - Enable은 ColorPicker를 활성화하고 Button 레이블을 Hide로 변경합니다.
   - Hide는 ColorPicker를 보이지 않게 하고 Button 레이블을 Show로 변경합니다.
5. [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

Button 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 ActionScript를 사용하여 전환 Button을 만들고 사용자가 Button을 클릭할 때 [출력] 패널에 이벤트 유형을 표시합니다. 이 예제에서는 클래스 생성자를 호출하여 Button 인스턴스를 만들고 addChild() 메서드를 호출하여 스테이지에 추가합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 Button 구성 요소를 현재 문서의 [라이브러리] 패널로 드래그합니다.

구성 요소가 라이브러리에 추가되고 응용 프로그램에는 표시되지 않습니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력하여 Button 인스턴스를 만듭니다.

```
import fl.controls.Button;

var aButton:Button = new Button();
addChild(aButton);
aButton.label = "Click me";
aButton.toggle =true;
aButton.move(50, 50);
```

move() 메서드는 스테이지의 50(x 좌표), 50(y 좌표) 위치에 버튼을 배치합니다.

4 이제, 다음 ActionScript를 추가하여 이벤트 리스너 및 이벤트 핸들러 함수를 만듭니다.

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    trace("Event type: " + event.type);
}
```

5 [컨트롤] > [무비 테스트]를 선택합니다.

버튼을 클릭하면 출력 패널에 "Event type: click"이라는 메시지가 표시됩니다.

CheckBox 구성 요소 사용

CheckBox는 선택하거나 선택을 해제할 수 있는 정사각형 상자입니다. 선택하면 상자에 체크 표시가 나타납니다. CheckBox에 텍스트 레이블을 추가한 후 CheckBox 왼쪽, 오른쪽, 위 또는 아래에 배치할 수 있습니다.

CheckBox를 사용하여 동시에 선택할 수 있는 true 또는 false 값 집합을 만들 수 있습니다. 예를 들어, 한 응용 프로그램에서 CheckBox를 사용하여 사용자가 구매하려는 차종 정보를 선택하도록 할 수 있습니다.

CheckBox와 사용자의 상호 작용

응용 프로그램에서 CheckBox를 활성화하거나 비활성화할 수 있습니다. CheckBox가 활성화되어 있는 상태에서 사용자가 상자 또는 레이블을 클릭하면 CheckBox가 입력 포커스를 받고 눌린 모양으로 표시됩니다. 사용자가 마우스 버튼을 누른 상태로 CheckBox 또는 해당 레이블의 경계 영역 밖으로 포인터를 이동하면 구성 요소의 모양은 원래 상태로 되돌아가고 입력 포커스는 유지됩니다. CheckBox의 상태는 마우스 포인터가 구성 요소에서 벗어날 때까지 변경되지 않습니다. 또한 CheckBox에는 selected와 deselected라는 두 가지 비활성 상태가 있는데, 이러한 상태는 각각 마우스나 키보드 상호 작용을 허용하지 않는 selectedDisabledSkin과 disabledSkin을 사용합니다.

CheckBox가 비활성화된 경우에는 사용자 상호 작용에 관계없이 비활성화된 모양으로 표시됩니다. CheckBox가 비활성화된 상태에서는 마우스 또는 키보드 입력을 수신하지 못합니다.

사용자가 CheckBox 인스턴스를 클릭하거나 Tab 키를 눌러 선택하면 해당 인스턴스로 포커스가 이동합니다. CheckBox 인스턴스에 포커스가 있을 때는 다음 키를 사용하여 해당 인스턴스를 제어할 수 있습니다.

키	설명
Shift+Tab	이전 요소로 포커스를 이동합니다.
스페이스바	구성 요소를 선택하거나 선택 취소하고 change 이벤트를 트리거합니다.
탭	다음 요소로 포커스를 이동합니다.

포커스 제어에 대한 자세한 내용은 24페이지의 “FocusManager를 사용한 작업”과 Flash Professional용 **ActionScript 3.0 참조 설명서**의 FocusManager 클래스를 참조하십시오.

각 CheckBox 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다.

응용 프로그램에 CheckBox 구성 요소를 추가할 때 다음 ActionScript 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

구성 요소의 인스턴스 수에 관계없이 구성 요소의 액세스 가능성을 한 번만 활성화하면 됩니다.

CheckBox 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 CheckBox 구성 요소 인스턴스에 대해 label, labelPlacement, selected 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 CheckBox 클래스를 참조하십시오.

CheckBox를 사용하여 응용 프로그램 만들기

다음 절차에서는 대출 응용 프로그램 양식을 예로 들어 응용 프로그램을 제작하는 동안 CheckBox 구성 요소를 추가하는 방법을 설명합니다. 이 양식은 신청자가 자택 소유자인지 묻고 "yes"라고 대답할 수 있는 CheckBox를 제공합니다. 자택을 소유한 경우 양식에는 자택의 상대적인 가격을 나타낼 수 있는 두 개의 라디오 버튼이 표시됩니다.

CheckBox 구성 요소를 사용하여 응용 프로그램 만들기

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널의 CheckBox 구성 요소를 스테이지로 드래그합니다.
3. 속성 관리자에서 다음을 수행합니다.
   - 인스턴스 이름으로 **homeCh**를 입력합니다.
   - 폭(W) 값으로 **140**을 입력합니다.
   - 레이블 매개 변수로 "**Own your home?**"을 입력합니다.
4. [구성 요소] 패널에서 두 개의 RadioButton 구성 요소를 스테이지로 드래그하여 CheckBox 아래쪽과 오른쪽에 배치합니다. 속성 관리자에서 다음 RadioButton 값을 입력합니다.
   - 인스턴스 이름으로 **underRb** 및 **overRb**를 입력합니다.
   - 두 RadioButton의 W(폭) 매개 변수로 **120**을 입력합니다.
   - **underRb** 레이블 매개 변수로 Under $500,000?를 입력합니다.
   - **overRb** 레이블 매개 변수로 Over $500,000?를 입력합니다.
   - 두 RadioButton에 groupName 매개 변수로 **valueGrp**를 입력합니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
homeCh.addEventListener(MouseEvent.CLICK, clickHandler);
underRb.enabled = false;
overRb.enabled = false;

function clickHandler(event:MouseEvent):void {
    underRb.enabled = event.target.selected;
    overRb.enabled = event.target.selected;
}
```

이 코드에서는 CLICK 이벤트에 대해 homeCh CheckBox가 선택된 경우 underRb 및 overRb RadioButton을 활성화하고 homeCh가 선택되지 않은 경우에는 둘 다 비활성화하는 이벤트 핸들러를 만듭니다. 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**에서 MouseEvent 클래스를 참조하십시오.

6 [컨트롤] > [무비 테스트]를 선택합니다.

다음 예제에서는 이전 응용 프로그램과 동일한 응용 프로그램을 만들지만 이번에는 ActionScript를 사용하여 CheckBox와 RadioButton을 만듭니다.

ActionScript를 사용하여 Checkbox 만들기

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 CheckBox 구성 요소와 RadioButton 구성 요소를 현재 문서의 [라이브러리] 패널로 드래그합니다. [라이브러리] 패널이 열려 있지 않으면 Ctrl+L을 누르거나 [윈도우] > [라이브러리]를 선택하여 [라이브러리] 패널을 엽니다.

이렇게 하면 응용 프로그램에서 구성 요소를 사용할 수 있지만 구성 요소가 스테이지에 배치되지는 않습니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력하여 구성 요소 인스턴스를 만들고 배치합니다.

```
import fl.controls.CheckBox;
import fl.controls.RadioButton;

var homeCh:CheckBox = new CheckBox();
var underRb:RadioButton = new RadioButton();
var overRb:RadioButton = new RadioButton();
addChild(homeCh);
addChild(underRb);
addChild(overRb);
underRb.groupName = "valueGrp";
overRb.groupName = "valueGrp";
homeCh.move(200, 100);
homeCh.width = 120;
homeCh.label = "Own your home?";
underRb.move(220, 130);
underRb.enabled = false;
underRb.width = 120;
underRb.label = "Under $500,000?";
overRb.move(220, 150);
overRb.enabled = false;
overRb.width = 120;
overRb.label = "Over $500,000?";
```

이 코드에서는 CheckBox() 생성자와 RadioButton() 생성자를 사용하여 구성 요소를 만들고 addChild() 메서드를 사용하여 스테이지에 해당 구성 요소를 배치합니다. 또한 move() 메서드를 사용하여 구성 요소를 스테이지에 배치합니다.

4 이제, 다음 ActionScript를 추가하여 이벤트 리스너 및 이벤트 핸들러 함수를 만듭니다.

```
homeCh.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    underRb.enabled = event.target.selected;
    overRb.enabled = event.target.selected;
}
```

이 코드는 CLICK 이벤트에 대해 homeCh CheckBox가 선택된 경우 underRb 및 overRb 라디오 버튼을 활성화하고 homeCh가 선택되지 않은 경우에는 두 라디오 버튼을 비활성화하는 이벤트 핸들러를 만듭니다. 자세한 내용은 Flash Professional용 ActionScript 3.0 참조 설명서에서 MouseEvent 클래스를 참조하십시오.

5 [컨트롤] > [무비 테스트]를 선택합니다.

ColorPicker 구성 요소 사용

ColorPicker 구성 요소를 사용하면 사용자가 견본 목록에서 색상을 선택할 수 있습니다. ColorPicker의 기본 모드에서는 사각형 버튼에 단일 색상이 표시됩니다. 사용자가 버튼을 클릭하면 견본 패널에 사용 가능한 색상 목록이 나타나고 현재 선택한 색상의 16진수 값을 표시하는 텍스트 필드가 나타납니다.

ColorPicker의 colors 속성을 원하는 색상 값으로 설정하여 ColorPicker에 나타나는 색상을 설정할 수 있습니다.

ColorPicker 구성 요소와 사용자의 상호 작용

ColorPicker를 사용하면 응용 프로그램에서 사용자가 색상을 선택하여 다른 객체에 적용할 수 있습니다. 예를 들어, 사용자가 배경색이나 텍스트 색상 같은 응용 프로그램 요소를 원하는 방식으로 설정할 수 있게 하려면 ColorPicker를 포함시키고 사용자가 선택한 색상을 적용하면 됩니다.

사용자는 패널에서 견본을 클릭하거나 텍스트 필드에 16진수 값을 입력하여 색상을 선택합니다. 사용자가 색상을 선택하면 ColorPicker의 selectedColor 속성을 사용하여 응용 프로그램의 텍스트나 다른 객체에 색상을 적용할 수 있습니다.

사용자가 포인터를 ColorPicker 위로 이동하거나 Tab 키를 눌러 ColorPicker를 선택하면 ColorPicker 인스턴스로 포커스가 이동합니다. ColorPicker의 견본 패널이 열려 있으면 다음 키를 사용하여 ColorPicker를 제어할 수 있습니다.

키	설명
홈	견본 패널에서 첫 번째 색상으로 이동합니다.
위쪽 화살표	견본 패널에서 한 행 위로 이동합니다.
아래쪽 화살표	견본 패널에서 한 행 아래로 이동합니다.
오른쪽 화살표	견본 패널에서 오른쪽에 있는 색상으로 이동합니다.
왼쪽 화살표	견본 패널에서 왼쪽에 있는 색상으로 이동합니다.
End	견본 패널에서 마지막 색상으로 이동합니다.

ColorPicker 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 ColorPicker 인스턴스에 대해 selectedColor, showTextField 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 ActionScript 3.0 Reference for Flash Professional에서 ColorPicker 클래스를 참조하십시오.

ColorPicker 구성 요소를 사용하여 응용 프로그램 만들기

다음 예제에서는 응용 프로그램을 제작하는 동안 ColorPicker 구성 요소를 추가합니다. 이 예제에서는 ColorPicker에서 색상을 변경할 때마다 changeHandler() 함수가 drawBox() 함수를 호출하여 ColorPicker에서 선택한 색상으로 새 상자를 그립니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ColorPicker를 스테이지 가운데로 드래그하고 인스턴스 이름을 **aCp**로 지정합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.events.ColorPickerEvent;

var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xFF0000);//draw a red box
addChild(aBox);

aCp.addEventListener(ColorPickerEvent.CHANGE,changeHandler);

function changeHandler(event:ColorPickerEvent):void {
    drawBox(aBox, event.target.selectedColor);
}

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1);
            box.graphics.drawRect(100, 150, 100, 100);
            box.graphics.endFill();
}
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

5 ColorPicker를 클릭하고 색상을 선택하여 상자에 색상을 지정합니다.

ActionScript를 사용하여 ColorPicker 만들기

다음 예제에서는 ColorPicker() 생성자와 addChild()를 사용하여 스테이지에 ColorPicker를 만들고 colors 속성을 빨강(0xFF0000), 녹색(0x00FF00), 파랑(0x0000FF)의 색상 값으로 설정하여 ColorPicker에 표시할 색상을 지정합니다. 또한 TextArea를 만들고 ColorPicker에서 다른 색상을 선택할 때마다 TextArea의 텍스트 색상이 선택한 색상과 일치하도록 변경합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ColorPicker 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [구성 요소] 패널의 TextArea 구성 요소를 [라이브러리] 패널로 드래그합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.ColorPicker;
import fl.controls.TextArea;
import fl.events.ColorPickerEvent;

var aCp:ColorPicker = new ColorPicker();
var aTa:TextArea = new TextArea();
var aTf:TextFormat = new TextFormat();

aCp.move(100, 100);
aCp.colors = [0xff0000, 0x00ff00, 0x0000ff];
aCp.addEventListener(ColorPickerEvent.CHANGE, changeHandler);

aTa.text = "Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Vivamus quis nisl vel tortor nonummy
vulputate. Quisque sit amet eros sed purus euismod tempor. Morbi tempor. Class aptent taciti sociosqu ad
litora torquent per conubia nostra, per inceptos hymenaeos. Curabitur diam. Suspendisse at purus in ipsum
volutpat viverra. Nulla pellentesque libero id libero.";
aTa.setSize(200, 200);
aTa.move(200,100);

addChild(aCp);
addChild(aTa);

function changeHandler(event:ColorPickerEvent):void {
    if(TextFormat(aTa.getStyle("textFormat"))){
        aTf = TextFormat(aTa.getStyle("textFormat"));
    }
    aTf.color = event.target.selectedColor;
    aTa.setStyle("textFormat", aTf);
}
```

5 [컨트롤] > [무비 테스트]를 선택합니다.

ComboBox 구성 요소 사용

ComboBox 구성 요소를 사용하면 사용자가 드롭다운 목록에서 한 항목을 선택할 수 있습니다. ComboBox는 정적일 수도 있고 편집 가능할 수도 있습니다. 편집 가능한 ComboBox를 사용하면 사용자가 목록 위에 있는 텍스트 필드에 직접 텍스트를 입력할 수 있습니다. 드롭 다운 목록이 너무 길어서 아래쪽으로 표시될 수 없으면 위로 열립니다. ComboBox는 BaseButton, TextInput 및 List 구성 요소의 하위 구성 요소로 이루어집니다.

편집 가능한 ComboBox에서는 버튼만 히트 영역이고 텍스트 상자는 히트 영역이 아닙니다. 정적 ComboBox에서는 버튼과 텍스트 상자가 히트 영역입니다. 히트 영역은 드롭 다운 목록을 열거나 닫아서 응답합니다.

사용자가 마우스나 키보드를 사용하여 목록에서 선택을 하면 선택 항목의 레이블이 ComboBox 맨 위에 있는 텍스트 필드에 복사됩니다.

ComboBox 구성 요소와 사용자의 상호 작용

목록에서 한 항목만 선택해야 하는 양식이나 응용 프로그램에서는 ComboBox 구성 요소를 사용합니다. 예를 들어, 고객 주소 양식에는 지역을 표시하는 드롭 다운 목록을 사용할 수 있습니다. 복잡한 시나리오에서는 편집 가능한 ComboBox를 사용합니다. 예를 들어, 운전 방향을 알려 주는 응용 프로그램에는 사용자가 출발지와 목적지 주소를 입력할 수 있게 편집 가능한 ComboBox를 사용할 수 있습니다. 드롭 다운 목록에는 이전에 사용자가 입력한 주소가 표시됩니다.

ComboBox를 편집할 수 있는 경우, 즉 editable 속성이 true로 설정되어 있는 경우에는 다음 키를 사용하여 텍스트 입력 상자의 포커스를 제거하고 이전 값을 유지할 수 있습니다. 사용자가 텍스트를 입력한 경우 새 값을 먼저 적용하는 Enter 키는 예외입니다.

키	설명
Shift+Tab	이전 항목으로 포커스를 이동합니다. 새 항목이 선택된 경우에는 change 이벤트가 전달됩니다.
탭	다음 항목으로 포커스를 이동합니다. 새 항목이 선택된 경우에는 change 이벤트가 전달됩니다.
아래쪽 화살표	선택이 한 항목 아래로 이동합니다.
End	목록의 아래쪽으로 이동합니다.
Escape	드롭 다운 목록을 닫고 ComboBox로 포커스를 되돌립니다.
Enter	드롭 다운 목록을 닫고 ComboBox로 포커스를 되돌립니다. ComboBox를 편집할 수 있는 경우 사용자가 텍스트를 입력하고 Enter 키를 누르면 해당 값이 입력한 텍스트로 설정됩니다.
홈	선택이 목록 맨 위로 이동합니다.
Page Up	선택이 한 페이지 위로 이동합니다.
Page Down	선택이 한 페이지 아래로 이동합니다.

응용 프로그램에 ComboBox 구성 요소를 추가할 때 다음 ActionScript 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.ComboBoxAccImpl;

ComboBoxAccImpl.enableAccessibility();
```

구성 요소의 인스턴스 수에 관계없이 구성 요소의 액세스 가능성을 한 번만 활성화하면 됩니다.

ComboBox 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 ComboBox 인스턴스에 대해 dataProvider, editable, prompt, rowCount 등의 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 ComboBox 클래스를 참조하십시오. dataProvider 매개 변수 사용에 대한 자세한 내용은 26페이지의 “dataProvider 매개 변수 사용”을 참조하십시오.

ComboBox 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램을 제작하는 동안 ComboBox 구성 요소를 추가하는 방법을 설명합니다. ComboBox가 편집 가능할 때 텍스트 필드에 **Add**를 입력하면 드롭 다운 목록에 항목이 추가됩니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. ComboBox 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aCb**로 지정합니다. [매개 변수] 탭에서 editable 매개 변수를 true로 설정합니다.
3. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력합니다.

```
import fl.data.DataProvider;
import fl.events.ComponentEvent;

var items:Array = [
{label:"screen1", data:"screenData1"},
{label:"screen2", data:"screenData2"},
{label:"screen3", data:"screenData3"},
{label:"screen4", data:"screenData4"},
{label:"screen5", data:"screenData5"},
];
aCb.dataProvider = new DataProvider(items);

aCb.addEventListener(ComponentEvent.ENTER, onAddItem);

function onAddItem(event:ComponentEvent):void {
    var newRow:int = 0;
    if (event.target.text == "Add") {
        newRow = event.target.length + 1;
            event.target.addItemAt({label:"screen" + newRow, data:"screenData" + newRow},
            event.target.length);
    }
}
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

ActionScript를 사용하여 ComboBox 만들기

다음 예제에서는 ActionScript로 ComboBox를 만들어 미국 캘리포니아주 샌프란시스코 지역에 있는 대학 목록으로 채웁니다. 그런 다음 ComboBox의 width 속성을 프롬프트 텍스트 폭에 맞는 크기로 설정하고 dropdownWidth 속성을 가장 긴 대학 이름보다 약간 넓게 설정합니다.

이 예제에서는 학교 이름을 저장하는 데 label 속성을, 학교의 웹 사이트 URL을 저장하는 데 data 속성을 사용하여 Array 인스턴스에 대학 목록을 만듭니다. 그런 다음 ComboBox의 dataProvider 속성을 설정하여 ComboBox에 Array를 지정합니다.

사용자가 목록에서 대학을 선택하면 Event.CHANGE 이벤트가 트리거되고 changeHandler() 함수가 호출되어 data 속성이 URL 요청에 로드되므로 결과적으로 학교의 웹 사이트에 액세스할 수 있게 됩니다.

마지막 행에서는 ComboBox 인스턴스의 selectedIndex 속성을 -1로 설정하여 목록이 닫힐 때 확인 대화 상자가 다시 표시되게 합니다. 이 행이 없으면 선택한 학교의 이름이 대신 표시됩니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ComboBox 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.ComboBox;
import fl.data.DataProvider;
import flash.net.navigateToURL;

var sfUniversities:Array = new Array(
    {label:"University of California, Berkeley",
                data:"http://www.berkeley.edu/"},
    {label:"University of San Francisco",
                data:"http://www.usfca.edu/"},
    {label:"San Francisco State University",
                data:"http://www.sfsu.edu/"},
    {label:"California State University, East Bay",
                data:"http://www.csuhayward.edu/"},
    {label:"Stanford University", data:"http://www.stanford.edu/"},
    {label:"University of Santa Clara", data:"http://www.scu.edu/"},
    {label:"San Jose State University", data:"http://www.sjsu.edu/"}
);

var aCb:ComboBox = new ComboBox();
aCb.dropdownWidth = 210;
aCb.width = 200;
aCb.move(150, 50);
aCb.prompt = "San Francisco Area Universities";
aCb.dataProvider = new DataProvider(sfUniversities);
aCb.addEventListener(Event.CHANGE, changeHandler);

addChild(aCb);

function changeHandler(event:Event):void {
    var request:URLRequest = new URLRequest();
    request.url = ComboBox(event.target).selectedItem.data;
    navigateToURL(request);
    aCb.selectedIndex = -1;
}
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

Flash 제작 환경에서 이 예제를 구현하고 실행할 수는 있지만 ComboBox에서 항목을 클릭하여 대학 웹 사이트에 액세스하려고 하면 경고 메시지가 나타납니다. 인터넷에서 기능을 제대로 갖춘 ComboBox에 액세스하려면 다음 위치에 액세스하십시오.

http://www.helpexamples.com/peter/bayAreaColleges/bayAreaColleges.html

DataGrid 구성 요소 사용

DataGrid 구성 요소를 사용하면 DataProvider의 배열로 파싱할 수 있는 외부 XML 파일이나 배열에서 데이터를 가져와 행과 열로 이루어진 격자에 표시할 수 있습니다. DataGrid 구성 요소에는 가로/세로 스크롤 기능, 이벤트 지원(편집 가능한 셀 지원 포함), 정렬 기능 등이 있습니다.

격자의 열 테두리, 색상, 글꼴 같은 특징의 크기를 조절하고 사용자 정의할 수 있습니다. 격자의 열에 대해 사용자 정의 무비 클립을 셀 렌더러로 사용할 수 있습니다. 셀 렌더러는 셀의 내용을 표시합니다. 스크롤 막대를 해제하고 DataGrid 메서드를 사용하여 페이지 뷰 스타일 디스플레이를 만들 수도 있습니다. 사용자 정의에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**에서 DataGridColumn 클래스를 참조하십시오.

기타 도움말 항목

Creating, populating, and resizing the DataGrid component

Customizing and sorting the DataGrid component

Filtering and formatting data in the DataGrid component

DataGrid 구성 요소와 사용자의 상호 작용

마우스 및 키보드를 사용하여 DataGrid 구성 요소와 상호 작용할 수 있습니다.

sortableColumns 속성과 열의 sortable 속성이 모두 true로 설정되어 있는 경우 열 머리글을 클릭하면 열 값을 기준으로 데이터가 정렬됩니다. 개별 열의 sortable 속성을 false로 설정하여 정렬 기능을 해제할 수 있습니다.

resizableColumns 속성이 true이면 머리글 행에서 열 구분자를 드래그하여 열 크기를 조절할 수 있습니다.

편집할 수 있는 셀을 클릭하면 포커스가 해당 셀로 이동합니다. 편집할 수 없는 셀을 클릭하면 포커스에 아무런 영향을 미치지 않습니다. 개별 셀의 DataGrid.editable 및 DataGridColumn.editable 속성이 모두 true이면 셀을 편집할 수 있습니다.

자세한 내용은 **ActionScript 3.0 참조 설명서**에서 **DataGrid** 및 **DataGridColumn** 클래스를 참조하십시오.

DataGrid 인스턴스에 포커스가 있으면 다음 키를 누르거나 클릭하여 인스턴스를 제어할 수 있습니다.

키	설명
아래쪽 화살표	셀을 편집 중이면 삽입점이 셀의 텍스트 끝으로 이동합니다. 셀을 편집할 수 없는 경우 아래쪽 화살표 키를 누르면 List 구성 요소에서처럼 선택 내용이 처리됩니다.
위쪽 화살표	셀을 편집 중이면 삽입점이 셀의 텍스트 시작 지점으로 이동합니다. 셀을 편집할 수 없는 경우 위쪽 화살표 키를 누르면 List 구성 요소에서처럼 선택 내용이 처리됩니다.
Shift+Up/아래쪽 화살표	DataGrid를 편집할 수 없고 allowMultipleSelection이 true이면 연속된 행이 선택됩니다. 반대쪽 화살표로 방향을 바꾸면 시작 행에 도달할 때까지 선택된 행의 선택이 취소되고, 시작 행부터 반대 방향의 행이 선택됩니다.
Shift+클릭	allowMultipleSelection이 true이면 선택된 행과 현재 캐럿 위치(강조 표시된 셀) 사이에 있는 모든 행이 선택됩니다.
Ctrl+클릭	allowMultipleSelection이 true이면 행이 추가로 선택됩니다. 행이 연속적일 필요는 없습니다.
오른쪽 화살표	셀을 편집 중이면 삽입점이 한 글자 오른쪽으로 이동합니다. 셀을 편집할 수 없는 경우에는 오른쪽 화살표 키를 눌러도 아무런 작업이 수행되지 않습니다.
왼쪽 화살표	셀을 편집 중이면 삽입점이 한 글자 왼쪽으로 이동합니다. 셀을 편집할 수 없는 경우에는 왼쪽 화살표 키를 눌러도 아무런 작업이 수행되지 않습니다.
홈	DataGrid에서 첫 번째 행을 선택합니다.
End	DataGrid에서 마지막 행을 선택합니다.
PageUp	DataGrid의 페이지에서 첫 번째 행을 선택합니다. 페이지는 DataGrid가 스크롤 없이 표시할 수 있는 행 수로 구성됩니다.

키	설명
PageDown	DataGrid의 페이지에서 마지막 행을 선택합니다. 페이지는 DataGrid가 스크롤 없이 표시할 수 있는 행 수로 구성됩니다.
Return/Enter/Shift+Enter	셀을 편집할 수 있는 경우 변경 내용이 커밋되고 삽입점이 동일한 열의 다음 행(Shift 키를 누른 상태인지에 따라 위쪽 또는 아래쪽 행)에 있는 셀로 이동합니다.
Shift+Tab/Tab	DataGrid가 편집 가능하면 열 끝에 도달할 때까지 포커스를 이전/다음 항목으로 옮기고 열 끝에 도달한 후에는 첫 번째 셀 또는 마지막 셀에 도달할 때까지 이전/다음 행으로 포커스를 옮깁니다. 첫 번째 셀이 선택된 상태에서 Shift+Tab을 누르면 포커스가 이전 컨트롤로 이동합니다. 마지막 셀이 선택된 상태에서 Tab 키를 누르면 포커스가 다음 컨트롤로 이동합니다. DataGrid를 편집할 수 없으면 포커스가 이전/다음 컨트롤로 이동합니다.

DataGrid 구성 요소를 여러 유형의 데이터 기반 응용 프로그램의 기초로 사용할 수 있습니다. 서식이 지정된 표 형식의 데이터 뷰를 쉽게 표시할 수 있고 셀 렌더러 기능을 사용하여 보다 정교하고 편집 가능한 사용자 인터페이스를 만들 수도 있습니다. 다음은 DataGrid 구성 요소를 실제로 활용한 예입니다.

- 웹 메일 클라이언트
- 검색 결과 페이지
- 대출 계산기 및 세금 양식 응용 프로그램 같은 스프레드시트 응용 프로그램

DataGrid 클래스는 SelectableList 클래스를 확장한 것이므로, DataGrid 구성 요소로 응용 프로그램을 디자인할 때 List 구성 요소의 디자인을 잘 알고 있으면 도움이 됩니다. SelectableList 클래스와 List 구성 요소에 대한 자세한 내용은 **ActionScript 3.0 참조 설명서**에서 **SelectableList** 및 **List** 클래스를 참조하십시오.

응용 프로그램에 DataGrid 구성 요소를 추가할 때 다음 ActionScript 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.DataGridAccImpl;
DataGridAccImpl.enableAccessibility();
```

구성 요소의 인스턴스 수에 관계없이 구성 요소의 액세스 가능성을 한 번만 활성화하면 됩니다. 자세한 내용은 **Flash 사용 설명서**에서 18장, "액세스 가능한 내용 만들기"를 참조하십시오.

DataGrid 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 DataGrid 구성 요소 인스턴스에 대해 allowMultipleSelection, editable, headerHeight, horizontalLineScrollSize, horizontalPageScrollSize, horizontalScrolllPolicy, resizableColumns, rowHeight, showHeaders, verticalLineScrollSize, verticalPageScrollSize, verticalScrollPolicy 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 DataGrid 클래스를 참조하십시오.

DataGrid 구성 요소를 사용하여 응용 프로그램 만들기

DataGrid 구성 요소를 사용하여 응용 프로그램을 만들려면 사용할 데이터의 출처를 먼저 결정해야 합니다. 일반적으로 데이터는 Array에 있으며 dataProvider 속성을 설정하여 이를 격자로 가져올 수 있습니다. 또한 DataGrid 및 DataGridColumn 클래스의 메서드를 사용하여 격자에 데이터를 추가할 수도 있습니다.

DataGrid 구성 요소에서 로컬 데이터 공급자 사용

다음 예제에서는 소프트볼 팀의 선수 명단을 표시하는 DataGrid를 만듭니다. 또한 선수 명단을 Array(aRoster)에 정의하고 이를 DataGrid의 dataProvider 속성에 지정합니다.

1 Flash에서 [파일] > [새로 만들기]를 선택하고 [Flash 파일(ActionScript 3.0)]을 선택합니다.

2 [구성 요소] 패널의 DataGrid 구성 요소를 스테이지로 드래그합니다.

3 속성 관리자에서 인스턴스 이름으로 **aDg**를 입력합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.data.DataProvider;

bldRosterGrid(aDg);
var aRoster:Array = new Array();
aRoster = [
        {Name:"Wilma Carter", Bats:"R", Throws:"R", Year:"So", Home: "Redlands, CA"},
        {Name:"Sue Pennypacker", Bats:"L", Throws:"R", Year:"Fr", Home: "Athens, GA"},
        {Name:"Jill Smithfield", Bats:"R", Throws:"L", Year:"Sr", Home: "Spokane, WA"},
        {Name:"Shirley Goth", Bats:"R", Throws:"R", Year:"Sr", Home: "Carson, NV"},
        {Name:"Jennifer Dunbar", Bats:"R", Throws:"R", Year:"Fr", Home: "Seaside, CA"},
        {Name:"Patty Crawford", Bats:"L", Throws:"L", Year:"Jr", Home: "Whittier, CA"},
        {Name:"Angelina Davis", Bats:"R", Throws:"R", Year:"So", Home: "Odessa, TX"},
        {Name:"Maria Santiago", Bats:"L", Throws:"L", Year:"Sr", Home: "Tacoma, WA"},
        {Name:"Debbie Ferguson", Bats:"R", Throws:"R", Year: "Jr", Home: "Bend, OR"},
        {Name:"Karen Bronson", Bats:"R", Throws:"R", Year: "Sr", Home: "Billings, MO"},
        {Name:"Sylvia Munson", Bats:"R", Throws:"R", Year: "Jr", Home: "Pasadena, CA"},
        {Name:"Carla Gomez", Bats:"R", Throws:"L", Year: "Sr", Home: "Corona, CA"},
        {Name:"Betty Kay", Bats:"R", Throws:"R", Year: "Fr", Home: "Palo Alto, CA"},
];
aDg.dataProvider = new DataProvider(aRoster);
aDg.rowCount = aDg.length;

function bldRosterGrid(dg:DataGrid){
    dg.setSize(400, 300);
    dg.columns = ["Name", "Bats", "Throws", "Year", "Home"];
    dg.columns[0].width = 120;
    dg.columns[1].width = 50;
    dg.columns[2].width = 50;
    dg.columns[3].width = 40;
    dg.columns[4].width = 120;
    dg.move(50,50);
};
```

bldRosterGrid() 함수는 DataGrid 크기를 설정하고 열 순서 및 크기를 설정합니다.

5 [컨트롤] > [무비 테스트]를 선택합니다.

응용 프로그램에서 DataGrid 구성 요소에 대한 열 지정 및 정렬 추가

열 머리글을 클릭하면 해당 열 값을 기준으로 DataGrid 내용을 내림차순으로 정렬할 수 있습니다.

다음 예제에서는 addColumn() 메서드를 사용하여 DataGrid에 DataGridColumn 인스턴스를 추가합니다. 열에는 선수 이름과 득점이 표시됩니다. 이 예제에서는 sortOptions 속성을 설정하여 각 열의 정렬 옵션도 지정합니다. 즉, Name 열에는 Array.CASEINSENSITIVE를 지정하고 Score 열에는 Array.NUMERIC을 지정합니다. 또한 길이를 행 수에 맞게 설정하고 폭을 200으로 설정하여 DataGrid 크기를 조절합니다.

1 Flash에서 [파일] > [새로 만들기]를 선택하고 [Flash 파일(ActionScript 3.0)]을 선택합니다.

2 [구성 요소] 패널의 DataGrid 구성 요소를 스테이지로 드래그합니다.

3 속성 관리자에서 인스턴스 이름으로 **aDg**를 입력합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.events.DataGridEvent;
import fl.data.DataProvider;
// Create columns to enable sorting of data.
var nameDGC:DataGridColumn = new DataGridColumn("name");
nameDGC.sortOptions = Array.CASEINSENSITIVE;
var scoreDGC:DataGridColumn = new DataGridColumn("score");
scoreDGC.sortOptions = Array.NUMERIC;
aDg.addColumn(nameDGC);
aDg.addColumn(scoreDGC);
var aDP_array:Array = new Array({name:"clark", score:3135}, {name:"Bruce", score:403}, {name:"Peter",
score:25})
aDg.dataProvider = new DataProvider(aDP_array);
aDg.rowCount = aDg.length;
aDg.width = 200;
```

5 [컨트롤] > [무비 테스트]를 선택합니다.

ActionScript를 사용하여 DataGrid 구성 요소 인스턴스 만들기

다음 예제에서는 ActionScript를 사용하여 DataGrid를 만들고 이를 선수 이름과 득점 Array로 채웁니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 DataGrid 구성 요소를 현재 문서의 [라이브러리] 패널로 드래그합니다.

구성 요소가 라이브러리에 추가되고 응용 프로그램에는 표시되지 않습니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aDg:DataGrid = new DataGrid();
addChild(aDg);
aDg.columns = [ "Name", "Score" ];
aDg.setSize(140, 100);
aDg.move(10, 40);
```

이 코드에서는 DataGrid 인스턴스를 만든 후 격자 크기를 조절하고 격자를 배치합니다.

4 배열을 만들어 데이터를 추가하고 DataGrid에 대한 데이터 공급자로 식별합니다.

```
var aDP_array:Array = new Array();
aDP_array.push({Name:"Clark", Score:3135});
aDP_array.push({Name:"Bruce", Score:403});
aDP_array.push({Name:"Peter", Score:25});
aDg.dataProvider = new DataProvider(aDP_array);
aDg.rowCount = aDg.length;
```

5 [컨트롤] > [무비 테스트]를 선택합니다.

XML 파일로 DataGrid 로드

다음 예제에서는 DataGridColumn 클래스를 사용하여 DataGrid 열을 만듭니다. 그런 다음 XML 객체를 DataProvider() 생성자의 value 매개 변수로 전달하여 DataGrid를 채웁니다.

1 텍스트 편집기를 사용하여 다음 데이터가 포함된 XML 파일을 만든 후 FLA 파일을 저장하는 폴더와 동일한 폴더에 team.xml로 저장합니다.

```
<team>
    <player name="Player A" avg="0.293" />
    <player name="Player B" avg="0.214" />
    <player name="Player C" avg="0.317" />
</team>
```

2 새 Flash(ActionScript 3.0) 문서를 만듭니다.

3 [구성 요소] 패널의 DataGrid 구성 요소를 두 번 클릭하여 스테이지에 추가합니다.

4 속성 관리자에서 인스턴스 이름으로 **aDg**를 입력합니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;
import flash.net.*;
import flash.events.*;

var request:URLRequest = new URLRequest("team.xml");
var loader:URLLoader = new URLLoader;


loader.load(request);
loader.addEventListener(Event.COMPLETE, loaderCompleteHandler);

function loaderCompleteHandler(event:Event):void {

    var teamXML:XML = new XML(loader.data);

    var nameCol:DataGridColumn = new DataGridColumn("name");
    nameCol.headerText = "Name";
    nameCol.width = 120;
    var avgCol:DataGridColumn = new DataGridColumn("avg");
    avgCol.headerText = "Average";
    avgCol.width = 60;

    var myDP:DataProvider = new DataProvider(teamXML);

    aDg.columns = [nameCol, avgCol];
    aDg.width = 200;
    aDg.dataProvider = myDP;
    aDg.rowCount = aDg.length;
}
```

6 [컨트롤] > [무비 테스트]를 선택합니다.

Label 구성 요소 사용

단일 텍스트 행을 표시하는 Label 구성 요소는 일반적으로 웹 페이지의 다른 요소 또는 작업을 식별하기 위해 텍스트 행 하나를 표시합니다. HTML을 사용하여 레이블 서식을 지정하면 HTML의 텍스트 서식 태그를 이용할 수 있습니다. 레이블의 정렬과 크기를 제어할 수도 있습니다. Label 구성 요소는 테두리가 없으며 포커스를 받을 수 없고 이벤트를 브로드캐스팅하지 않습니다.

각 Label 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다. 레이블에는 테두리가 없으므로 text 매개 변수를 설정해야 실시간 미리 보기를 볼 수 있습니다.

Label 구성 요소와 사용자의 상호 작용

Label 구성 요소를 사용하여 양식의 다른 구성 요소에 대한 텍스트 레이블(예: 사용자 이름이 입력되는TextInput 필드 왼쪽의 "Name:" 레이블)을 만듭니다. 일반 텍스트 필드 대신 스타일을 사용하여 일관된 모양과 느낌을 유지할 수 있는 Label 구성 요소를 사용하는 것이 좋습니다.

Label 구성 요소를 회전하려면 글꼴을 포함해야 합니다. 그렇지 않으면 무비를 테스트할 때 레이블이 표시되지 않습니다.

Label 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 Label 구성 요소 인스턴스에 대해 autoSize, condenseWhite, selectable, text, wordWrap 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 Label 클래스를 참조하십시오.

Label 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 제작하는 동안 응용 프로그램에 Label 구성 요소를 추가하는 방법을 설명합니다. 이 예제는 레이블로 "만료일 " 텍스트를 표시합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 Label 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 지정합니다.

- 인스턴스 이름으로 **aLabel**을 입력합니다.
- W 값으로 **80**을 입력합니다.
- X 값으로 **100**를 입력합니다.
- Y 값으로 **100**을 입력합니다.
- text 매개 변수로 **만료일**을 입력합니다.

3 TextArea 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 지정합니다.

- 인스턴스 이름으로 **aTa**를 입력합니다.
- H 값으로 **22**를 입력합니다.
- X 값으로 **200**를 입력합니다.
- Y 값으로 **100**을 입력합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
var today:Date = new Date();
var expDate:Date = addDays(today, 14);
aTa.text = expDate.toDateString();

function addDays(date:Date, days:Number):Date {
return addHours(date, days*24);
}

function addHours(date:Date, hrs:Number):Date {
return addMinutes(date, hrs*60);
}

function addMinutes(date:Date, mins:Number):Date {
return addSeconds(date, mins*60);
}

function addSeconds(date:Date, secs:Number):Date {
var mSecs:Number = secs * 1000;
var sum:Number = mSecs + date.getTime();
return new Date(sum);
}
```

5 [컨트롤] > [무비 테스트]를 선택합니다.

ActionScript를 사용하여 Label 구성 요소 인스턴스 만들기

다음 예제에서는 ActionScript를 사용하여 Label 매개 변수를 만듭니다. Label을 사용하여 ColorPicker 구성 요소의 기능을 설명하고 htmlText 속성을 사용하여 Label의 텍스트에 서식을 적용합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 Label 구성 요소를 현재 문서의 [라이브러리] 패널로 드래그합니다.

3 [구성 요소] 패널의 ColorPicker 구성 요소를 현재 문서의 [라이브러리] 패널로 드래그합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.Label;
import fl.controls.ColorPicker;

var aLabel:Label = new Label();
var aCp:ColorPicker = new ColorPicker();

addChild(aLabel);
addChild(aCp);

aLabel.htmlText = '<font face="Arial" color="#FF0000" size="14">Fill:</font>';
aLabel.x = 200;
aLabel.y = 150;
aLabel.width = 25;
aLabel.height = 22;

aCp.x = 230;
aCp.y = 150;
```

5 [컨트롤] > [무비 테스트]를 선택합니다.

List 구성 요소 사용

List 구성 요소는 한 항목만 선택할 수 있거나 여러 항목을 선택할 수 있는 스크롤 가능한 목록 상자입니다. 목록에는 다른 구성 요소뿐만 아니라 그래픽도 표시할 수 있습니다. labels 또는 data 매개 변수 필드를 클릭하면 나타나는 [값] 대화 상자를 사용하여 목록에 표시되는 항목을 추가합니다. 또한 List.addItem() 및 List.addItemAt() 메서드를 사용하여 목록에 항목을 추가할 수 있습니다.

List 구성 요소는 0부터 시작하는 인덱스를 사용하며 인덱스가 0인 항목이 맨 위에 표시됩니다. List 클래스 메서드 및 속성을 사용하여 목록 항목을 추가, 제거 및 교체할 때 목록 항목의 인덱스를 지정해야 할 수도 있습니다.

List 구성 요소와 사용자의 상호 작용

사용자가 하나만 선택할 수 있거나 여러 개를 선택할 수 있도록 목록을 설정할 수 있습니다. 예를 들어, 전자 상거래 웹 사이트를 방문한 사용자가 구매할 항목을 선택할 경우, 목록에서 30개의 항목을 스크롤하여 확인한 후 한 항목을 클릭하여 선택합니다.

또한 사용자 정의 무비 클립을 행으로 사용하는 List를 디자인하여 사용자에 대한 자세한 정보를 표시할 수 있습니다. 예를 들어, 전자 메일 응용 프로그램에서 각 사서함은 List 구성 요소가 될 수 있고 각 행은 우선 순위 및 상태를 나타내는 아이콘일 수 있습니다.

목록을 클릭하거나 Tab 키를 눌러 선택하면 해당 목록으로 포커스가 이동합니다. 그러면 다음 키를 사용하여 목록을 제어할 수 있습니다.

키	설명
영숫자 키	레이블의 첫 문자가 Key.getAscii()인 다음 항목으로 이동합니다.
Ctrl	인접하지 않은 여러 항목을 선택하거나 선택 취소하는 데 사용되는 전환 키입니다.
아래쪽 화살표	한 항목 아래가 선택됩니다.
홈	목록의 맨 위가 선택됩니다.
Page Down	한 페이지 아래가 선택됩니다.
Page Up	한 페이지 위가 선택됩니다.
Shift	연속적으로 선택할 수 있습니다.
위쪽 화살표	한 항목 위가 선택됩니다.

참고: 스크롤 크기는 행 단위가 아니라 픽셀 단위입니다.

참고: Page Up 및 Page Down 키에서 사용되는 페이지 크기는 한 번에 표시되는 항목의 수보다 하나 작습니다. 예를 들어, 10행으로 된 드롭 다운 목록에서는 Page Down 키를 누를 때마다 항목이 0-9, 9-18, 18-27 등으로 페이지마다 한 항목이 겹쳐서 표시됩니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**의 IFocusManager 인터페이스 및 FocusManager 클래스와 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

스테이지에 있는 각 List 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다.

응용 프로그램에 List 구성 요소를 추가할 때 다음 ActionScript 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.ListAccImpl;

ListAccImpl.enableAccessibility();
```

구성 요소의 인스턴스 수에 관계없이 구성 요소의 액세스 가능성을 한 번만 활성화하면 됩니다. 자세한 내용은 **Flash 사용 설명서**에서 18장, "액세스 가능한 내용 만들기"를 참조하십시오.

List 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 List 구성 요소 인스턴스에 대해 allowMultipleSelection, dataProvider, horizontalLineScrollSize, horizontalPageScrollSize, horizontalScrollPolicy, multipleSelection, verticalLineScrollSize, verticalPageScrollSize, verticalScrollPolicy 등의 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 List 클래스를 참조하십시오. dataProvider 매개 변수 사용에 대한 자세한 내용은 26페이지의 "dataProvider 매개 변수 사용"을 참조하십시오.

List 구성 요소를 사용하여 응용 프로그램 만들기

다음 예제에서는 응용 프로그램을 제작하는 동안 List 구성 요소를 추가하는 방법을 설명합니다.

응용 프로그램에 단순 List 구성 요소 추가

이 예제에서 List는 차량 모델을 나타내는 레이블과 가격이 들어 있는 데이터 필드로 구성됩니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 List 구성 요소를 스테이지로 드래그합니다.

3 속성 관리자에서 다음을 수행합니다.

- 인스턴스 이름으로 **aList**를 입력합니다.
- W(폭) 값을 **200**으로 지정합니다.

4 [텍스트 도구]를 사용하여 aList 아래에 텍스트 필드를 만들고 인스턴스 이름을 **aTf**로 지정합니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.List;
import flash.text.TextField;

aTf.type = TextFieldType.DYNAMIC;
aTf.border = false;

// Create these items in the Property inspector when data and label
// parameters are available.
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.allowMultipleSelection = true;

aList.addEventListener(Event.CHANGE, showData);

function showData(event:Event) {
    aTf.text = "This car is priced at: $" + event.target.selectedItem.data;
}
```

이 코드에서는 addItem() 메서드를 사용하여 세 가지 항목으로 aList를 채우고, 목록에 표시되는 label 값과 data 값을 각 항목에 지정합니다. List에서 항목을 선택하면 이벤트 리스너가 showData() 함수를 호출하여 선택한 항목의 data 값을 표시합니다.

6 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 컴파일하고 실행합니다.

데이터 공급자로 List 인스턴스 채우기

다음 예제에서는 차량 모델과 가격으로 구성된 List를 만들고, addItem() 메서드 대신 데이터 공급자를 사용하여 List를 채웁니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 List 구성 요소를 스테이지로 드래그합니다.

3 속성 관리자에서 다음을 수행합니다.

- 인스턴스 이름으로 **aList**를 입력합니다.
- W(폭) 값을 **200**으로 지정합니다.

4 [텍스트 도구]를 사용하여 aList 아래에 텍스트 필드를 만들고 인스턴스 이름을 **aTf**로 지정합니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.List;
import fl.data.DataProvider;
import flash.text.TextField;

aTf.type = TextFieldType.DYNAMIC;
aTf.border = false;

var cars:Array = [
{label:"1956 Chevy (Cherry Red)", data:35000},
{label:"1966 Mustang (Classic)", data:27000},
{label:"1976 Volvo (Xcllnt Cond)", data:17000},
];
aList.dataProvider = new DataProvider(cars);
aList.allowMultipleSelection = true;

aList.addEventListener(Event.CHANGE, showData);

function showData(event:Event) {
    aTf.text = "This car is priced at: $" + event.target.selectedItem.data;
}
```

6 [컨트롤] > [무비 테스트]를 선택하여 List와 해당 항목을 봅니다.

List 구성 요소를 사용하여 MovieClip 인스턴스 제어

다음 예제에서는 색상 이름 List를 만들어 색상 이름 중 하나를 선택할 때 MovieClip에 해당 색상이 적용되도록 합니다.

1 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 List 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 지정합니다.

- 인스턴스 이름으로 **aList**를 입력합니다.
- H 값으로 **60**을 입력합니다.
- X 값으로 **100**를 입력합니다.
- Y 값으로 **150**을 입력합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
aList.addItem({label:"Blue", data:0x0000CC});
aList.addItem({label:"Green", data:0x00CC00});
aList.addItem({label:"Yellow", data:0xFFFF00});
aList.addItem({label:"Orange", data:0xFF6600});
aList.addItem({label:"Black", data:0x000000});

var aBox:MovieClip = new MovieClip();
addChild(aBox);

aList.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event) {
    drawBox(aBox, event.target.selectedItem.data);
};

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(225, 150, 100, 100);
            box.graphics.endFill();
}
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

5 List에서 색상을 클릭하여 MovieClip에 표시되는지 확인합니다.

ActionScript를 사용하여 List 구성 요소 인스턴스 만들기

다음 예제에서는 ActionScript를 사용하여 간단한 목록을 만들고 addItem() 메서드를 사용하여 이 목록을 채웁니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 List 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.List;

var aList:List = new List();
aList.addItem({label:"One", data:1});
aList.addItem({label:"Two", data:2});
aList.addItem({label:"Three", data:3});
aList.addItem({label:"Four", data:4});
aList.addItem({label:"Five", data:5});
aList.setSize(60, 40);
aList.move(200,200);
addChild(aList);
aList.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event):void {
    trace(event.target.selectedItem.data);
}
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

NumericStepper 구성 요소 사용

NumericStepper 구성 요소를 사용하면 사용자가 정렬된 숫자 집합에서 순서대로 이동할 수 있습니다. 이 구성 요소는 위쪽 및 아래쪽 화살표 버튼 옆에 표시되는 텍스트 상자의 숫자로 구성됩니다. 버튼을 누르면 버튼을 놓거나 최대값 또는 최소값에 도달할 때까지 stepSize 매개 변수에 지정된 단위에 따라 점증적으로 숫자가 올라가거나 내려갑니다. NumericStepper 구성 요소의 텍스트 상자에 있는 텍스트를 편집할 수도 있습니다.

각 NumericStepper 인스턴스의 실시간 미리 보기에는 속성 관리자나 [구성 요소 관리자]에서 지정한 값 매개 변수의 설정이 반영됩니다. 그러나 실시간 미리 보기의 NumericStepper 화살표 버튼으로 마우스 또는 키보드를 사용할 수는 없습니다.

NumericStepper 구성 요소와 사용자의 상호 작용

사용자가 숫자 값을 선택하도록 할 모든 위치에 NumericStepper 구성 요소를 사용할 수 있습니다. 예를 들어, 양식에 NumericStepper 구성 요소를 사용하여 신용 카드 만료일(년, 월, 일)을 설정할 수 있습니다. 사용자가 글꼴 크기를 늘리거나 줄이게 하는 데에도 NumericStepper 구성 요소를 사용합니다.

NumericStepper 구성 요소는 숫자 데이터만 처리합니다. 또한 제작하는 동안 스태퍼의 크기를 조절하여 세 자리 이상의 숫자가 표시되도록 해야 합니다(예: 5246 또는 1.34).

응용 프로그램에서 NumericStepper를 활성화하거나 비활성화할 수 있습니다. NumericStepper가 비활성화된 상태에서는 마우스 또는 키보드 입력을 수신하지 못합니다. 활성화된 상태에서 NumericStepper를 클릭하거나 Tab 키를 눌러 선택하면 NumericStepper로 포커스가 이동하고 텍스트 상자로 내부 포커스가 설정됩니다. NumericStepper 인스턴스가 포커스를 받으면 다음 키를 사용하여 해당 인스턴스를 제어할 수 있습니다.

키	설명
아래쪽 화살표	한 단위씩 값을 변경합니다.
왼쪽 화살표	텍스트 상자 안에서 삽입점을 왼쪽으로 이동합니다.
오른쪽 화살표	텍스트 상자 안에서 삽입점을 오른쪽으로 이동합니다.
Shift+Tab	이전 객체로 포커스를 이동합니다.
탭	다음 객체로 포커스를 이동합니다.
위쪽 화살표	한 단위씩 값을 변경합니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 ActionScript 3.0 참조 설명서의 FocusManager 클래스 및 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

NumericStepper 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 NumericStepper 인스턴스에 대해 maximum, minimum, stepSize, value 등의 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 NumericStepper 클래스를 참조하십시오.

NumericStepper를 사용하여 응용 프로그램 만들기

다음 절차에서는 제작하는 동안 응용 프로그램에 NumericStepper 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서는 NumericStepper 구성 요소와 Label 구성 요소를 스테이지에 배치하고, NumericStepper 인스턴스의 Event.CHANGE 이벤트에 대한 리스너를 만듭니다. NumericStepper의 값이 변경되면 Label 인스턴스의 text 속성에 새 값이 표시됩니다.

1. [구성 요소] 패널의 NumericStepper 구성 요소를 스테이지로 드래그합니다.
2. 속성 관리자에서 인스턴스 이름으로 **aNs**를 입력합니다.
3. [구성 요소] 패널의 Label 구성 요소를 스테이지로 드래그합니다.
4. 속성 관리자에서 인스턴스 이름으로 **aLabel**을 입력합니다.
5. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import flash.events.Event;

aLabel.text = "value = " + aNs.value;

aNs.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event) :void {
    aLabel.text = "value = " + event.target.value;
};
```

   이 예제에서는 레이블의 text 속성을 NumericStepper 값으로 설정합니다. changeHandler() 함수는 NumericStepper 인스턴스 값이 변경될 때마다 레이블의 text 속성을 업데이트합니다.

6. [컨트롤] > [무비 테스트]를 선택합니다.

ActionScript를 사용하여 NumericStepper 만들기

다음 예제에서는 ActionScript 코드를 사용하여 사용자 출생일의 년, 월, 일을 각각 입력하는 세 개의 NumericStepper를 만듭니다. 또한 각 NumericStepper의 식별자 및 프롬프트에 사용할 Label을 추가합니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. Label을 [라이브러리] 패널로 드래그합니다.

3 NumericStepper를 [라이브러리] 패널로 드래그합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.Label;
import fl.controls.NumericStepper;

var dobPrompt:Label = new Label();
var moPrompt:Label = new Label();
var dayPrompt:Label = new Label();
var yrPrompt:Label = new Label();

var moNs:NumericStepper = new NumericStepper();
var dayNs:NumericStepper = new NumericStepper();
var yrNs:NumericStepper = new NumericStepper();

addChild(dobPrompt);
addChild(moPrompt);
addChild(dayPrompt);
addChild(yrPrompt);
addChild(moNs);
addChild(dayNs);
addChild(yrNs);

dobPrompt.setSize(65, 22);
dobPrompt.text = "Date of birth:";
dobPrompt.move(80, 150);

moNs.move(150, 150);
moNs.setSize(40, 22);
moNs.minimum = 1;
moNs.maximum = 12;
moNs.stepSize = 1;
moNs.value = 1;

moPrompt.setSize(25, 22);
moPrompt.text = "Mo.";
moPrompt.move(195, 150);

dayNs.move(225, 150);
dayNs.setSize(40, 22);
dayNs.minimum = 1;
dayNs.maximum = 31;
dayNs.stepSize = 1;
dayNs.value = 1;

dayPrompt.setSize(25, 22);
dayPrompt.text = "Day";
dayPrompt.move(270, 150);

yrNs.move(300, 150);
yrNs.setSize(55, 22);
yrNs.minimum = 1900;
yrNs.maximum = 2006;
yrNs.stepSize = 1;
yrNs.value = 1980;

yrPrompt.setSize(30, 22);
yrPrompt.text = "Year";
yrPrompt.move(360, 150);
```

5 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

ProgressBar 구성 요소 사용

ProgressBar 구성 요소는 내용이 로드되는 진행률을 표시하여 내용이 커서 응용 프로그램 실행이 지연될 수 있을 때 사용자가 안심할 수 있도록 합니다. ProgressBar는 이미지와 응용 프로그램 일부의 로드 진행률을 표시하는 데 유용합니다. 로드 프로세스는 determinate 또는 indeterminate일 수 있습니다. **determinate** 진행률 막대는 시간에 따른 작업 진행률을 선형으로 표시하며 로드할 내용의 크기를 알고 있을 때 사용됩니다. **indeterminate** 진행률 막대는 로드할 내용의 크기를 알 수 없을 때 사용됩니다. Label 구성 요소를 추가하여 로드 진행률을 백분율로 표시할 수도 있습니다.

ProgressBar 구성 요소는 9-슬라이스 크기 조절을 사용하며 막대 스킨, 트랙 스킨 및 불확정 스킨을 포함합니다.

ProgressBar 구성 요소와 사용자의 상호 작용

ProgressBar 구성 요소는 세 가지 모드로 사용할 수 있습니다. 가장 일반적으로 사용되는 모드는 event 모드와 polled 모드입니다. 이러한 모드는 progress 및 complete 이벤트를 내보내거나(event 및 polled 모드), bytesLoaded 및 bytesTotal 속성을 표시하는(polled 모드) 로드 프로세스를 지정합니다. ProgressBar.setProgress() 메서드를 호출하면서 maximum, minimum 및 value 속성을 함께 설정하여 ProgressBar 구성 요소를 manual 모드로 사용할 수도 있습니다. indeterminate 속성을 설정하여, 소스의 크기를 알 수 없을 때 ProgressBar를 줄무늬 모양으로 채우거나(true) 소스의 크기를 알 수 있을 때 단색으로 채울(false) 수 있습니다.

ActionScript를 사용하거나 속성 관리자 또는 구성 요소 관리자의 mode 매개 변수를 통해 ProgressBar의 mode 속성을 설정하여 모드를 설정할 수 있습니다.

100,000개의 항목을 파싱하는 것처럼 ProgressBar를 사용하여 처리 상태를 나타내는 경우 이러한 프로세스를 단일 프레임 루프로 처리하면 화면을 다시 그리는 과정이 없기 때문에 ProgressBar의 업데이트 상황을 볼 수 없습니다.

ProgressBar 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 ProgressBar 인스턴스에 대해 direction, mode, source 등의 매개 변수를 설정할 수 있습니다. 이러한 매개 변수마다 같은 이름의 해당 ActionScript 속성이 있습니다.

ActionScript를 작성하면 속성, 메서드 및 이벤트를 사용하여 ProgressBar 구성 요소에 대한 이러한 옵션 및 추가 옵션을 제어할 수 있습니다. 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 ProgressBar 클래스를 참조하십시오.

ProgressBar를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램을 제작하는 동안 ProgressBar 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서 ProgressBar는 event 모드를 사용합니다. event 모드에서는 ProgressBar가 진행률을 표시하기 위해 전달하는 progress 및 complete 이벤트를 로드되고 있는 내용이 내보냅니다. progress 이벤트가 발생하면 로드된 내용의 백분율을 나타내는 레이블이 업데이트됩니다. complete 이벤트가 발생하면 "Loading complete"라는 메시지와 파일의 크기를 나타내는 bytesTotal 속성 값이 표시됩니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ProgressBar 구성 요소를 스테이지로 드래그합니다.

- 속성 관리자에서 인스턴스 이름으로 **aPb**를 입력합니다.
- [매개 변수] 섹션에서 X 값으로 **200**을 입력합니다.
- Y 값으로 **260**을 입력합니다.
- mode 매개 변수에 대해 event를 선택합니다.

3 [구성 요소] 패널의 Button 구성 요소를 스테이지로 드래그합니다.

- 속성 관리자에서 인스턴스 이름으로 **loadButton**을 입력합니다.

- X 매개 변수로 **220**을 입력합니다.
- Y 매개 변수로 **290**을 입력합니다.
- **label** 매개 변수에 사운드 로드를 입력합니다.

4 Label 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **progLabel**로 지정합니다.

- W 값으로 **150**을 입력합니다.
- X 매개 변수로 **200**을 입력합니다.
- Y 매개 변수로 **230**을 입력합니다.
- [매개 변수] 섹션에서 text 매개 변수 값을 지웁니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 .mp3 오디오 파일을 로드하는 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.ProgressBar;
import flash.events.ProgressEvent;
import flash.events.IOErrorEvent;

var aSound:Sound = new Sound();
aPb.source = aSound;
var url:String = "http://www.helpexamples.com/flash/sound/song1.mp3";
var request:URLRequest = new URLRequest(url);

aPb.addEventListener(ProgressEvent.PROGRESS, progressHandler);
aPb.addEventListener(Event.COMPLETE, completeHandler);
aSound.addEventListener(IOErrorEvent.IO_ERROR, ioErrorHandler);
loadButton.addEventListener(MouseEvent.CLICK, clickHandler);

function progressHandler(event:ProgressEvent):void {
        progLabel.text = ("Sound loading ... " + aPb.percentComplete);
}

function completeHandler(event:Event):void {
    trace("Loading complete");
    trace("Size of file: " + aSound.bytesTotal);
    aSound.close();
    loadButton.enabled = false;
}

function clickHandler(event:MouseEvent) {
    aSound.load(request);
}

function ioErrorHandler(event:IOErrorEvent):void {
    trace("Load failed due to: " + event.text);
}
```

6 [컨트롤] > [무비 테스트]를 선택합니다.

polled 모드에서 ProgressBar 구성 요소를 사용하여 응용 프로그램 만들기

다음 예제에서는 ProgressBar를 polled 모드로 설정합니다. polled 모드에서 진행률은 로드되고 있는 내용에 대한 progress 이벤트를 수신하고 bytesLoaded 및 bytesTotal 속성을 통해 진행률을 계산하여 결정됩니다. 이 예제에서는 Sound 객체를 로드하고 이 객체의 progress 이벤트를 수신한 후 bytesLoaded 및 bytesTotal 속성을 사용하여 로드되는 백분율을 계산합니다. 그런 다음 레이블과 [출력] 패널에 로드 진행률을 표시합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ProgressBar 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 입력합니다.

- 인스턴스 이름으로 **aPb**를 입력합니다.
- X 값으로 **185**를 입력합니다.
- Y 값으로 **225**을 입력합니다.

3 Label 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 입력합니다.

- 인스턴스 이름으로 **progLabel**을 입력합니다.
- X 값으로 **180**를 입력합니다.
- Y 값으로 **180**을 입력합니다.
- [매개 변수] 섹션에서 text 매개 변수 값을 지웁니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다. 이 코드는 Sound 객체(aSound)를 만들고 loadSound()를 호출하여 Sound 객체에 사운드를 로드합니다.

```
import fl.controls.ProgressBarMode;
import flash.events.ProgressEvent;
import flash.media.Sound;

var aSound:Sound = new Sound();
var url:String = "http://www.helpexamples.com/flash/sound/song1.mp3";
var request:URLRequest = new URLRequest(url);

aPb.mode = ProgressBarMode.POLLED;
aPb.source = aSound;
aSound.addEventListener(ProgressEvent.PROGRESS, loadListener);

aSound.load(request);

function loadListener(event:ProgressEvent) {
    var percentLoaded:int = event.target.bytesLoaded / event.target.bytesTotal * 100;
    progLabel.text = "Percent loaded: " + percentLoaded + "%";
    trace("Percent loaded: " + percentLoaded + "%");
}
```

5 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

manual 모드에서 ProgressBar 구성 요소를 사용하여 응용 프로그램 만들기

다음 예제에서는 ProgressBar를 manual 모드로 설정합니다. manual 모드에서는 setProgress() 메서드를 호출하여 수동으로 진행률을 설정하고 현재 값과 최대값을 제공하여 진행률 범위를 결정해야 합니다. manual 모드에서는 source 속성을 설정하지 않습니다. 이 예제에서는 최대값이 250인 NumericStepper 구성 요소를 사용하여 ProgressBar를 늘립니다. NumericStepper 값이 변하면 CHANGE 이벤트가 트리거되고 이벤트 핸들러(nsChangeHander)가 setProgress() 메서드를 호출하여 ProgressBar를 진행시킵니다. 또한 최대값을 바탕으로 완료 상태를 백분율로 표시합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ProgressBar 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 지정합니다.

- 인스턴스 이름으로 **aPb**를 입력합니다.
- X 값으로 **180**를 입력합니다.
- Y 값으로 **175**을 입력합니다.

3 NumericStepper 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 입력합니다.

- 인스턴스 이름으로 **aNs**를 입력합니다.
- X 값으로 **220**를 입력합니다.

- Y 값으로 **215**을 입력합니다.
- [매개 변수] 섹션에서 maximum 매개 변수로 **250**, minimum 매개 변수로 **0**, stepSize 매개 변수로 **1**, value 매개 변수로 **0**을 입력합니다.

4 Label 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 입력합니다.

- 인스턴스 이름으로 **progLabel**을 입력합니다.
- W 값으로 **150**을 입력합니다.
- X 값으로 **180**를 입력합니다.
- Y 값으로 **120**을 입력합니다.
- [매개 변수] 탭에서 Label의 text 매개 변수 값을 지웁니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력합니다.

```
import fl.controls.ProgressBarDirection;
import fl.controls.ProgressBarMode;
import flash.events.Event;

aPb.direction = ProgressBarDirection.RIGHT;
aPb.mode = ProgressBarMode.MANUAL;
aPb.minimum = aNs.minimum;
aPb.maximum = aNs.maximum;
aPb.indeterminate = false;

aNs.addEventListener(Event.CHANGE, nsChangeHandler);

function nsChangeHandler(event:Event):void {
    aPb.value = aNs.value;
    aPb.setProgress(aPb.value, aPb.maximum);
progLabel.text = "Percent of progress = " + int(aPb.percentComplete) + "%";
}
```

6 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

7 NumericStepper에서 위쪽 화살표를 클릭하여 ProgressBar를 진행시킵니다.

ActionScript를 사용하여 ProgressBar 만들기

다음 예제에서는 ActionScript를 사용하여 ProgressBar를 만듭니다. 또한 manual 모드로 ProgressBar를 만드는 이전 예제의 기능을 복제합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 ProgressBar 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 NumericStepper 구성 요소를 [라이브러리] 패널로 드래그합니다.

4 Label 구성 요소를 [라이브러리] 패널로 드래그합니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력합니다.

```
import fl.controls.ProgressBar;
import fl.controls.NumericStepper;
import fl.controls.Label;
import fl.controls.ProgressBarDirection;
import fl.controls.ProgressBarMode;
import flash.events.Event;

var aPb:ProgressBar = new ProgressBar();
var aNs:NumericStepper = new NumericStepper();
var progLabel:Label = new Label();

addChild(aPb);
addChild(aNs);
addChild(progLabel);

aPb.move(180,175);
aPb.direction = ProgressBarDirection.RIGHT;
aPb.mode = ProgressBarMode.MANUAL;

progLabel.setSize(150, 22);
progLabel.move(180, 150);
progLabel.text = "";

aNs.move(220, 215);
aNs.maximum = 250;
aNs.minimum = 0;
aNs.stepSize = 1;
aNs.value = 0;

aNs.addEventListener(Event.CHANGE, nsChangeHandler);

function nsChangeHandler(event:Event):void {
aPb.setProgress(aNs.value, aNs.maximum);
progLabel.text = "Percent of progress = " + int(aPb.percentComplete) + "%";
}
```

6 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

7 NumericStepper에서 위쪽 화살표를 클릭하여 ProgressBar를 진행시킵니다.

RadioButton 구성 요소 사용

RadioButton 구성 요소를 사용하면 사용자가 선택 항목 집합 내에서 하나를 선택하도록 강제할 수 있습니다. 이 구성 요소는 RadioButton 인스턴스가 두 개 이상 있는 그룹에 사용해야 합니다. 항상 그룹의 한 멤버만 선택할 수 있습니다. 그룹의 한 라디오 버튼을 선택하면 그룹에서 현재 선택된 라디오 버튼은 선택 취소됩니다. 라디오 버튼이 속한 그룹을 나타내려면 groupName 매개 변수를 설정합니다.

라디오 버튼은 많은 웹 양식 응용 프로그램에서 기본이 되는 요소입니다. 사용자가 옵션 그룹에서 한 항목만 선택하도록 하려면 라디오 버튼을 사용합니다. 예를 들어, 고객이 사용할 신용 카드를 묻는 양식에 라디오 버튼을 사용할 수 있습니다.

RadioButton 구성 요소와 사용자의 상호 작용

라디오 버튼은 활성화하거나 비활성화할 수 있습니다. 비활성화된 상태에서 라디오 버튼은 마우스 또는 키보드 입력을 받지 못합니다. 사용자가 RadioButton 구성 요소 그룹을 클릭하거나 Tab 키를 눌러 선택하면 선택된 라디오 버튼만 포커스를 받습니다. 그러면 사용자가 다음 키를 사용하여 해당 인스턴스를 제어할 수 있습니다.

키	설명
위쪽 화살표/왼쪽 화살표	라디오 버튼 그룹 안의 이전 라디오 버튼을 선택합니다.
아래쪽 화살표/오른쪽 화살표	라디오 버튼 그룹 안의 다음 라디오 버튼을 선택합니다.
탭	라디오 버튼 그룹에서 다음 구성 요소로 포커스를 이동합니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**의 IFocusManager 인스턴스 및 FocusManager 클래스와 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

스테이지에 있는 각 RadioButton 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다. 그러나 선택되지 않은 항목은 실시간 미리 보기에 표시되지 않습니다. 같은 그룹 안에서 두 라디오 버튼에 대해 selected 매개 변수를 true로 설정하면 런타임에는 마지막으로 만들어진 인스턴스만 선택된 것으로 표시되지만 현재는 두 라디오 버튼이 선택된 것으로 표시됩니다. 자세한 내용은 70페이지의 "RadioButton 구성 요소 매개 변수"을 참조하십시오.

응용 프로그램에 RadioButton 구성 요소를 추가할 때 다음 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.RadioButtonAccImpl;
RadioButtonAccImpl.enableAccessibility();
```

구성 요소의 인스턴스 수에 관계없이 구성 요소의 액세스 가능성을 한 번만 활성화하면 됩니다. 자세한 내용은 Flash 사용 설명서에서 18장, "액세스 가능한 내용 만들기"를 참조하십시오.

RadioButton 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 RadioButton 구성 요소 인스턴스에 대해 groupName, label, LabelPlacement, selected, value 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 RadioButton 클래스를 참조하십시오.

ActionScript를 작성하면 RadioButton 클래스의 메서드, 속성 및 이벤트를 사용하여 RadioButton 인스턴스에 대한 추가 옵션을 설정할 수 있습니다.

RadioButton 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 제작하는 동안 응용 프로그램에 RadioButton 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서는 RadioButtons를 사용하여 예/아니요로 답할 수 있는 질문을 표현합니다. RadioButton의 데이터는 TextArea에 표시됩니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널에서 두 개의 RadioButton 구성 요소를 스테이지로 드래그합니다.
3. 첫 번째 라디오 버튼을 선택합니다. 속성 관리자에서 인스턴스 이름을 **yesRb**로 지정하고 그룹 이름을 **rbGroup**으로 지정합니다.
4. 두 번째 라디오 버튼을 선택합니다. 속성 관리자에서 인스턴스 이름을 **noRb**로 지정하고 그룹 이름을 **rbGroup**으로 지정합니다.
5. [구성 요소] 패널의 TextArea 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aTa**로 지정합니다.
6. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
yesRb.label = "Yes";
yesRb.value = "For";
noRb.label = "No";
noRb.value = "Against";

yesRb.move(50, 100);
noRb.move(100, 100);
aTa.move(50, 30);
noRb.addEventListener(MouseEvent.CLICK, clickHandler);
yesRb.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    aTa.text = event.target.value;
}
```

7 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

ActionScript를 사용하여 RadioButton 만들기

다음 예제에서는 ActionScript를 사용하여 빨강, 파랑 및 녹색에 대한 세 개의 RadioButton을 만들고 회색 상자를 그립니다. 각 RadioButton의 value 속성은 해당 버튼과 연결된 색상의 16진수 값을 지정합니다. 사용자가 RadioButton 중 하나를 클릭하면 clickHandler() 함수가 drawBox()를 호출하여 해당 RadioButton의 value 속성을 상자의 색상으로 전달합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 RadioButton 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.RadioButton;
import fl.controls.RadioButtonGroup;

var redRb:RadioButton = new RadioButton();
var blueRb:RadioButton = new RadioButton();
var greenRb:RadioButton = new RadioButton();
var rbGrp:RadioButtonGroup = new RadioButtonGroup("colorGrp");

var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xCCCCCC);

addChild(redRb);
addChild(blueRb);
addChild(greenRb);
addChild(aBox);

redRb.label = "Red";
redRb.value = 0xFF0000;
blueRb.label = "Blue";
blueRb.value = 0x0000FF;
greenRb.label = "Green";
greenRb.value = 0x00FF00;
redRb.group = blueRb.group = greenRb.group = rbGrp;
redRb.move(100, 260);
blueRb.move(150, 260);
greenRb.move(200, 260);

rbGrp.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    drawBox(aBox, event.target.selection.value);
}

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(125, 150, 100, 100);
            box.graphics.endFill();
}
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 RadioButton 클래스를 참조하십시오.

ScrollPane 구성 요소 사용

로드할 내용이 대상 표시 영역보다 큰 경우에는 ScrollPane 구성 요소를 사용하여 내용을 모두 표시할 수 있습니다. 예를 들어, 응용 프로그램에서 작은 공간에 큰 이미지를 표시할 때 ScrollPane에 로드하면 됩니다. ScrollPane에는 무비 클립, JPEG, PNG, GIF 및 SWF 파일을 사용할 수 있습니다.

ScrollPane 및 UILoader 같은 구성 요소에는 내용 로드가 완료된 시점을 확인할 수 있게 해 주는 complete 이벤트가 있습니다. ScrollPane 또는 UILoader 구성 요소의 내용에 대한 속성을 설정하려면 complete 이벤트를 수신하여 이벤트 핸들러에서 속성을 설정합니다. 예를 들어 다음 코드에서는 Event.COMPLETE 이벤트의 리스너를 만들고 ScrollPane 내용의 alpha 속성을 0.5로 설정하는 이벤트 핸들러를 만듭니다.

```
function spComplete(event:Event):void{
aSp.content.alpha = .5;
}
aSp.addEventListener(Event.COMPLETE, spComplete);
```

ScrollPane에 내용을 로드할 때 위치를 지정하려면 0, 0(X 및 Y 좌표)과 같은 형식으로 위치를 지정해야 합니다. 예를 들어, 다음 코드에서는 상자가 0 위치에 그려지므로 ScrollPane이 올바르게 로드됩니다.

```
var box:MovieClip = new MovieClip();
box.graphics.beginFill(0xFF0000, 1);
box.graphics.drawRect(0, 0, 150, 300);
box.graphics.endFill();
aSp.source = box;//load ScrollPane
```

자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 ScrollPane 클래스를 참조하십시오.

ScrollPane 구성 요소와 사용자의 상호 작용

ScrollPane은 활성화하거나 비활성화할 수 있습니다. 비활성화된 ScrollPane은 마우스 또는 키보드 입력을 받지 못합니다. ScrollPane에 포커스가 있는 경우 사용자는 다음 키를 사용하여 ScrollPane을 제어할 수 있습니다.

키	설명
아래쪽 화살표	내용을 세로 방향으로 한 행만큼 위로 스크롤합니다.
위쪽 화살표	내용을 세로 방향으로 한 행만큼 아래로 스크롤합니다.
End	ScrollPane의 맨 아래쪽 내용으로 이동합니다.
왼쪽 화살표	내용을 가로 방향으로 한 행만큼 오른쪽으로 스크롤합니다.
오른쪽 화살표	내용을 가로 방향으로 한 행만큼 왼쪽으로 스크롤합니다.
홈	ScrollPane의 맨 위쪽 내용으로 이동합니다.
End	ScrollPane의 맨 아래쪽 내용으로 이동합니다.
PageDown	내용을 세로 방향으로 한 페이지만큼 위로 스크롤합니다.
PageUp	내용을 세로 방향으로 한 페이지만큼 아래로 스크롤합니다.

사용자는 마우스를 사용하여 ScrollPane의 내용 및 가로/세로 스크롤 막대와 상호 작용할 수 있습니다. scrollDrag 속성이 true로 설정되어 있으면 마우스를 사용하여 내용을 드래그할 수 있습니다. 내용 위에 나타나는 손 모양 포인터는 사용자가 내용을 드래그할 수 있음을 나타냅니다. 대부분의 다른 컨트롤과 달리 마우스 버튼을 누른 채로 있으면 뗄 때까지 작업이 수행됩니다. 내용에 유효한 탭 중지가 있는 경우 scrollDrag를 false로 설정해야 합니다. 그렇지 않으면 내용을 히트하는 모든 마우스 동작이 스크롤 드래그를 실행합니다.

ScrollPane 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 ScrollPane 인스턴스에 대해 horizontalLineScrollSize, horizontalPageScrollSize, horizontalScrollPolicy, scrollDrag, source, verticalLineScrollSize, verticalPageScrollSize, verticalScrollPolicy 등의 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 ScrollPane 클래스를 참조하십시오.

ActionScript를 작성하면 속성, 메서드 및 이벤트를 사용하여 ScrollPane 구성 요소에 대한 이러한 옵션 및 추가 옵션을 제어할 수 있습니다.

ScrollPane 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 제작하는 동안 응용 프로그램에 ScrollPane 구성 요소를 추가하는 방법을 보여 줍니다. 이 예제의 ScrollPane은 source 속성으로 지정된 경로에서 그림을 로드합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ScrollPane 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aSp**로 지정합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.events.ScrollEvent;

aSp.setSize(300, 200);

function scrollListener(event:ScrollEvent):void {
    trace("horizontalScPosition: " + aSp.horizontalScrollPosition +
     ", verticalScrollPosition = " + aSp.verticalScrollPosition);
};
aSp.addEventListener(ScrollEvent.SCROLL, scrollListener);

function completeListener(event:Event):void {
    trace(event.target.source + " has completed loading.");
};
// Add listener.
aSp.addEventListener(Event.COMPLETE, completeListener);

aSp.source = "http://www.helpexamples.com/flash/images/image1.jpg";
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

ActionScript를 사용하여 ScrollPane 인스턴스 만들기

다음 예제에서는 ScrollPane을 만들고 창 크기를 설정한 다음 source 속성을 사용하여 이미지를 로드합니다. 또한 두 개의 리스너를 만듭니다. 첫 번째 리스너는 scroll 이벤트를 수신하고, 사용자가 세로 또는 가로로 스크롤할 때 이미지의 위치를 표시합니다. 두 번째 리스너는 complete 이벤트를 수신하고, 이미지 로드가 완료되었다는 메시지를 [출력] 패널에 표시합니다.

다음 예제에서는 ActionScript를 사용하여 ScrollPane을 만들고 그 안에 폭이 150픽셀이고 높이가 300픽셀인 MovieClip(빨간색 상자)을 배치합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ScrollPane 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [구성 요소] 패널의 DataGrid 구성 요소를 [라이브러리] 패널로 드래그합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.containers.ScrollPane;
import fl.controls.ScrollPolicy;
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aSp:ScrollPane = new ScrollPane();
var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xFF0000);//draw a red box

aSp.source = aBox;
aSp.setSize(150, 200);
aSp.move(100, 100);

addChild(aSp);

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1);
            box.graphics.drawRect(0, 0, 150, 300);
            box.graphics.endFill();
}
```

5 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

Slider 구성 요소 사용

Slider 구성 요소를 사용하면 사용자가 값 범위에 해당하는 트랙의 끝점 사이에 있는 그래픽 썸을 조정하여 값을 선택할 수 있습니다. 슬라이더를 사용하여 숫자, 백분율 등의 값을 선택할 수 있습니다. 또한 ActionScript를 사용하여 슬라이더의 값에 따라 두 번째 객체의 비헤이비어를 제어할 수 있습니다. 예를 들어, 슬라이더를 그림과 연결하고 슬라이더 썸의 상대 위치나 값에 따라 그림을 확대하거나 축소할 수 있습니다.

Slider의 현재 값은 트랙의 양쪽 끝 위치나 Slider의 최소값과 최대값 사이에서 썸의 상대적인 위치로 결정됩니다.

Slider에서는 최소값과 최대값 사이의 연속적인 값 범위를 사용할 수 있지만 snapInterval 매개 변수를 설정하여 최소값과 최대값 사이의 간격을 지정할 수도 있습니다. Slider에 트랙을 따라 지정된 간격으로 눈금 표시를 나타낼 수 있습니다. 이 값은 슬라이더에 지정된 값과 독립적입니다.

슬라이더는 기본적으로 가로 방향이지만 direction 매개 변수를 vertical로 설정하여 세로 방향으로 만들 수도 있습니다. 슬라이더 트랙은 양쪽 끝까지 확장되고 눈금 표시는 트랙 바로 위에 왼쪽에서 오른쪽으로 배치됩니다.

Slider 구성 요소와 사용자의 상호 작용

Slider 인스턴스에 포커스가 있으면 다음 키를 사용하여 해당 인스턴스를 제어할 수 있습니다.

키	설명
오른쪽 화살표	가로 슬라이더와 연결된 값을 높입니다.
위쪽 화살표	세로 슬라이더와 연결된 값을 높입니다.
왼쪽 화살표	가로 슬라이더와 연결된 값을 낮춥니다.
아래쪽 화살표	세로 슬라이더와 연결된 값을 낮춥니다.
Shift+Tab	이전 객체로 포커스를 이동합니다.
탭	다음 객체로 포커스를 이동합니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**의 IFocusManager 인터페이스 및 FocusManager 클래스와 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

각 Slider 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다.

Slider 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 Slider 구성 요소 인스턴스에 대해 direction, liveDragging, maximum, minimum, snapInterval, tickInterval, value 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 Slider 클래스를 참조하십시오.

Slider를 사용하여 응용 프로그램 만들기

다음 예제에서는 일종의 가상 이벤트에 대한 사용자 만족도를 표현할 수 있는 Slider 인스턴스를 만듭니다. 사용자는 Slider를 오른쪽이나 왼쪽으로 이동하여 만족도 수준을 높이거나 낮춥니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 Label 구성 요소를 스테이지 가운데로 드래그합니다.

- 인스턴스 이름을 **valueLabel**로 지정합니다.

- text 매개 변수에 **0percent** 값을 지정합니다.

3 [구성 요소] 패널의 Slider 구성 요소를 value_lbl 아래 가운데에 오도록 드래그합니다.

- 인스턴스 이름을 **aSlider**로 지정합니다.
- 폭(W:)을 **200**으로 지정합니다.
- 높이(H:)를 **10**으로 지정합니다.
- **maximum** 매개 변수에 100 값을 지정합니다.
- snapInterval 및 tickInterval 매개 변수에 **10** 값을 지정합니다.

4 [라이브러리] 패널의 다른 Label 인스턴스를 드래그하여 aSlider 아래의 가운데에 놓습니다.

- 인스턴스 이름을 **promptLabel**로 지정합니다.
- 폭(W:)을 250으로 지정합니다.
- 높이(H:)를 22로 지정합니다.
- **text** 매개 변수에 Please indicate your level of satisfaction을 입력합니다.

5 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.Slider;
import fl.events.SliderEvent;
import fl.controls.Label;

aSlider.addEventListener(SliderEvent.CHANGE, changeHandler);

function changeHandler(event:SliderEvent):void {
    valueLabel.text = event.value + "percent";
}
```

6 [컨트롤] > [무비 테스트]를 선택합니다.

이 예제에서는 슬라이더 썸을 다른 간격으로 이동하면 SliderEvent.CHANGE 이벤트의 리스너가 valueLabel의 text 속성을 업데이트하여 해당 썸 위치의 백분율을 표시합니다.

ActionScript를 사용하여 Slider 구성 요소가 있는 응용 프로그램 만들기

다음 예제에서는 ActionScript를 사용하여 Slider를 만듭니다. 또한 꽃 이미지를 다운로드하고 사용자가 Slider의 값에 해당하는 alpha 속성을 변경하여 이미지를 어둡거나 밝게 만들 수 있도록 합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 Label 구성 요소와 Slider 구성 요소를 현재 문서의 [라이브러리] 패널로 드래그합니다.

이렇게 하면 구성 요소가 라이브러리에 추가되지만 응용 프로그램에는 표시되지 않습니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력하여 구성 요소 인스턴스를 만들고 배치합니다.

```
import fl.controls.Slider;
import fl.events.SliderEvent;
import fl.controls.Label;
import fl.containers.UILoader;

var sliderLabel:Label = new Label();
sliderLabel.width = 120;
sliderLabel.text = "< Fade - Brighten >";
sliderLabel.move(170, 350);

var aSlider:Slider = new Slider();
aSlider.width = 200;
aSlider.snapInterval = 10;
aSlider.tickInterval = 10;
aSlider.maximum = 100;
aSlider.value = 100;
aSlider.move(120, 330);

var aLoader:UILoader = new UILoader();
aLoader.source = "http://www.flash-mx.com/images/image1.jpg";
aLoader.scaleContent = false;

addChild(sliderLabel);
addChild(aSlider);
addChild(aLoader);

aLoader.addEventListener(Event.COMPLETE, completeHandler);

function completeHandler(event:Event) {
    trace("Number of bytes loaded: " + aLoader.bytesLoaded);
}

aSlider.addEventListener(SliderEvent.CHANGE, changeHandler);

function changeHandler(event:SliderEvent):void {
        aLoader.alpha = event.value * .01;
}
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

5 슬라이더 썸을 왼쪽으로 이동하면 이미지가 어두워지고 오른쪽으로 이동하면 이미지가 밝아집니다.

TextArea 구성 요소 사용

TextArea 구성 요소는 기본 ActionScript TextField 객체의 래퍼입니다. TextArea 구성 요소를 사용하여 텍스트를 표시할 수 있으며, editable 속성이 true인 경우에는 텍스트를 입력하고 편집할 수도 있습니다. 이 구성 요소는 여러 행의 텍스트를 표시하거나 받을 수 있으며 wordWrap 속성이 true로 설정된 경우 긴 텍스트를 줄 바꿈할 수 있습니다. restrict 속성을 사용하면 사용자가 입력할 수 있는 문자를 제한할 수 있으며 maxChars를 사용하면 사용자가 입력할 수 있는 최대 문자 수를 지정할 수 있습니다. horizontalScrollPolicy 및 verticalScrollPolicy 속성이 off가 아닌 경우 텍스트가 텍스트 영역의 가로 또는 세로 경계를 초과하면 자동으로 가로 또는 세로 스크롤 막대가 나타납니다.

여러 행 텍스트 필드가 필요한 모든 위치에 TextArea 구성 요소를 사용할 수 있습니다. 예를 들어, 양식에서 TextArea 구성 요소를 주석 필드로 사용할 수 있습니다. 이 경우 사용자가 Tab 키를 눌러 필드 밖으로 이동할 때 필드가 비어 있는지 확인하는 리스너를 설정할 수 있습니다. 이 리스너는 필드가 비어 있으면 해당 필드에 주석을 입력해야 한다는 오류 메시지를 표시합니다.

한 행으로 된 텍스트 필드가 필요하면 TextInput 구성 요소를 사용하십시오.

TextArea 인스턴스에 나타나는 텍스트의 스타일을 변경하는 setStyle() 메서드를 사용하여 textFormat 스타일을 설정할 수 있습니다. 또한 ActionScript에서 htmlText 속성을 사용하면 TextArea 구성 요소에 HTML 서식을 지정할 수 있으며 displayAsPassword 속성을 true로 설정하면 텍스트를 별표로 가릴 수 있습니다. condenseWhite 속성을 true로 설정하면 새 텍스트에서 공백, 줄 바꿈 등과 같은 불필요한 공백이 제거됩니다. 이 속성은 컨트롤의 기존 텍스트에는 영향을 주지 않습니다.

TextArea 구성 요소와 사용자의 상호 작용

응용 프로그램 내에서 TextArea 구성 요소를 활성화하거나 비활성화할 수 있습니다. 비활성화된 상태에서 TextArea는 마우스 또는 키보드 입력을 수신하지 못합니다. 활성화된 상태에서는 ActionScript TextField 객체와 마찬가지로 포커스, 선택 및 탐색 규칙을 따릅니다. TextArea 인스턴스에 포커스가 있으면 다음 키를 사용하여 해당 인스턴스를 제어할 수 있습니다.

키	설명
화살표 키	텍스트를 편집할 수 있는 경우 텍스트 내에서 삽입점을 위, 아래, 왼쪽 또는 오른쪽으로 이동합니다.
Page Down	텍스트를 편집할 수 있는 경우 삽입점을 텍스트 끝으로 이동합니다.
Page Up	텍스트를 편집할 수 있는 경우 삽입점을 텍스트 처음으로 이동합니다.
Shift+Tab	탭 루프의 이전 객체로 포커스를 이동합니다.
탭	탭 루프의 다음 객체로 포커스를 이동합니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 ActionScript 3.0 참조 설명서의 FocusManager 클래스 및 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

TextArea 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 TextArea 구성 요소 인스턴스에 대해 condenseWhite, editable, hortizontalScrollPolicy, maxChars, restrict, text, verticalScrollPolicy, wordwrap 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 TextArea 클래스를 참조하십시오.

각 TextArea 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 구성 요소 관리자에서 변경한 매개 변수가 반영됩니다. 스크롤 막대가 필요한 경우에는 스크롤 막대가 실시간 미리 보기에 표시되지만 작동하지는 않습니다. 실시간 미리 보기에서는 텍스트를 선택할 수 없으며 스테이지에서 구성 요소 인스턴스에 텍스트를 입력할 수도 없습니다.

ActionScript를 작성하면 속성, 메서드 및 이벤트를 사용하여 TextArea 구성 요소에 대한 이러한 옵션 및 추가 옵션을 제어할 수 있습니다. 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 TextArea 클래스를 참조하십시오.

TextArea 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램을 제작하는 동안 TextArea 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서는 인터페이스의 다른 영역으로 포커스를 이동하기 전에 사용자가 텍스트 영역에 내용을 입력했는지 확인하는 focusOut 이벤트 핸들러를 TextArea 인스턴스에 설정합니다.

1. 새 Flash 문서(ActionScript 3.0)를 만듭니다.
2. [구성 요소] 패널의 TextArea 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aTa**로 지정합니다. 매개 변수 설정은 기본 설정을 유지합니다.
3. [구성 요소] 패널에서 두 번째 TextArea 구성 요소를 스테이지로 드래그하여 첫 번째 구성 요소 아래에 배치하고 인스턴스 이름을 **bTa**로 지정합니다. 매개 변수 설정은 기본 설정을 유지합니다.
4. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import flash.events.FocusEvent;

aTa.restrict = "a-z,'\" \"";
aTa.addEventListener(Event.CHANGE,changeHandler);
aTa.addEventListener(FocusEvent.KEY_FOCUS_CHANGE, k_m_fHandler);
aTa.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, k_m_fHandler);

function changeHandler(ch_evt:Event):void {
    bTa.text = aTa.text;
}
function k_m_fHandler(kmf_event:FocusEvent):void {
    kmf_event.preventDefault();
}
```

이 예제에서는 aTa 텍스트 영역에 입력할 수 있는 문자를 소문자, 쉼표, 아포스트로피 및 공백으로 제한합니다. 또한 aTa 텍스트 영역에서 change, KEY_FOCUS_CHANGE 및 MOUSE_FOCUS_CHANGE 이벤트에 대한 이벤트 핸들러를 설정합니다. changeHandler() 함수는 change 이벤트가 발생할 때마다 aTa.text를 bTa.text에 지정하여 aTa 텍스트 영역에 입력한 텍스트가 자동으로 bTa 텍스트 영역에 나타나게 합니다. KEY_FOCUS_CHANGE 및 MOUSE_FOCUS_CHANGE 이벤트의 k_m_fHandler() 함수는 아무 텍스트도 입력하지 않은 상태에서 Tab 키를 눌러 다음 필드로 이동하지 못하게 합니다. 이를 위해 기본 비헤이비어를 동작하지 않게 합니다.

5 [컨트롤] > [무비 테스트]를 선택합니다.

아무 텍스트도 입력하지 않고 Tab 키를 눌러 포커스를 두 번째 텍스트 영역으로 이동하면 오류 메시지가 나타나고 포커스가 첫 번째 텍스트 영역으로 돌아가야 합니다. 첫 번째 텍스트 영역에 텍스트를 입력하면 두 번째 텍스트 영역에 해당 텍스트가 복사되어야 합니다.

ActionScript를 사용하여 TextArea 인스턴스 만들기

다음 예제에서는 ActionScript를 사용하여 TextArea 구성 요소를 만듭니다. 또한 condenseWhite 속성을 true로 설정하여 불필요한 공백을 제거한 후 HTML 텍스트 서식 특성을 이용할 수 있도록 htmlText에 텍스트를 지정합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 TextArea 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.TextArea;

var aTa:TextArea = new TextArea();

aTa.move(100,100);
aTa.setSize(200, 200);
aTa.condenseWhite = true;
aTa.htmlText = '<b>Lorem ipsum dolor</b> sit amet, consectetuer adipiscing elit. <u>Vivamus quis nisl
vel tortor nonummy vulputate.</u> Quisque sit amet eros sed purus euismod tempor. Morbi tempor. <font
color="#FF0000">Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos
hymenaeos.</font> Curabitur diam. Suspendisse at purus in ipsum volutpat viverra. Nulla pellentesque
libero id libero.';
addChild(aTa);
```

이 예제에서는 htmlText 속성을 사용하여 텍스트 블록에 HTML 굵은 글꼴 및 밑줄 특성을 적용하고 a_ta 텍스트 영역에 해당 텍스트 블록을 표시합니다. 또한 condenseWhite 속성을 true로 설정하여 텍스트 블록 내의 필요 없는 공백을 제거합니다. setSize() 메서드는 텍스트 영역의 높이와 폭을 설정하고 move() 메서드는 텍스트 영역의 위치를 설정합니다. addChild() 메서드는 TextArea 인스턴스를 스테이지에 추가합니다.

4 [컨트롤] > [무비 테스트]를 선택합니다.

TextInput 구성 요소 사용

TextInput 구성 요소는 기본 ActionScript TextField 객체의 래퍼인 단일 행 텍스트 구성 요소입니다. 여러 행으로 된 텍스트 필드가 필요하면 TextArea 구성 요소를 사용합니다. 예를 들어, 양식에서 TextInput 구성 요소를 암호 필드로 사용할 수 있습니다. 이 경우 사용자가 Tab 키를 눌러 필드 밖으로 이동할 때 필드의 문자 수가 충분한지 확인하는 리스너를 설정할 수 있습니다. 이 리스너는 필드의 문자 수가 충분하지 않은 경우에 해당 필드에 적절한 수의 문자를 입력해야 한다는 오류 메시지를 표시합니다.

TextInput 인스턴스에 나타나는 텍스트의 속성을 변경하는 setStyle() 메서드를 사용하여 textFormat 속성을 설정할 수 있습니다. TextInput 구성 요소에는 텍스트를 숨기는 암호 필드 또는 HTML 형식을 설정할 수 있습니다.

TextInput 구성 요소와 사용자의 상호 작용

응용 프로그램 내에서 TextInput 구성 요소를 활성화하거나 비활성화할 수 있습니다. 비활성화된 상태에서 TextInput은 마우스 또는 키보드 입력을 수신하지 못합니다. 활성화된 상태에서는 ActionScript TextField 객체와 마찬가지로 포커스, 선택 및 탐색 규칙을 따릅니다. TextInput 인스턴스에 포커스가 있으면 다음 키를 사용하여 해당 인스턴스를 제어할 수 있습니다.

키	설명
화살표 키	삽입점을 왼쪽 또는 오른쪽으로 한 문자씩 이동합니다.
Shift+Tab	이전 객체로 포커스를 이동합니다.
탭	다음 객체로 포커스를 이동합니다.

포커스 제어에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**의 FocusManager 클래스 및 24페이지의 "FocusManager를 사용한 작업"을 참조하십시오.

각 TextInput 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다. 실시간 미리 보기에서는 텍스트를 선택할 수 없으며 스테이지에서 구성 요소 인스턴스에 텍스트를 입력할 수도 없습니다.

응용 프로그램에 TextInput 구성 요소를 추가할 때는 [액세스 가능성] 패널을 사용하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있도록 만들 수 있습니다.

TextInput 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 TextInput 구성 요소 인스턴스에 대해 editable, displayAsPassword, maxChars, restrict, text 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 TextInput 클래스를 참조하십시오.

ActionScript를 작성하면 속성, 메서드 및 이벤트를 사용하여 TextInput 구성 요소에 대한 이러한 옵션 및 추가 옵션을 제어할 수 있습니다. 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 TextInput 클래스를 참조하십시오.

TextInput 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램에 TextInput 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서는 두 개의 TextInput 필드를 사용하여 암호를 입력하고 확인합니다. 또한 이벤트 리스너를 사용하여 최소 8자가 입력되었는지, 두 필드의 텍스트가 일치하는지 확인합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 Label 구성 요소를 스테이지로 드래그하고 속성 관리자에서 다음 값을 지정합니다.

- 인스턴스 이름으로 **pwdLabel**을 입력합니다.

- W 값으로 **100**을 입력합니다.
- X 값으로 **50**을 입력합니다.
- Y 값으로 **150**을 입력합니다.
- [매개 변수] 섹션에서 text 매개 변수 값으로 **Password:**를 입력합니다.

3 [구성 요소] 패널에서 두 번째 Label 구성 요소를 스테이지로 드래그하고 다음 값을 지정합니다.

- 인스턴스 이름으로 **confirmLabel**을 입력합니다.
- W 값으로 **100**을 입력합니다.
- X 값으로 **50**을 입력합니다.
- Y 값으로 **200**을 입력합니다.
- [매개 변수] 섹션에서 text 매개 변수 값으로 **ConfirmPassword:**를 입력합니다.

4 [구성 요소] 패널의 TextInput 구성 요소를 스테이지로 드래그하고 다음 값을 지정합니다.

- 인스턴스 이름으로 **pwdTi**를 입력합니다.
- W 값으로 **150**을 입력합니다.
- X 값으로 **190**을 입력합니다.
- Y 값으로 **150**을 입력합니다.
- [매개 변수] 섹션에서 displayAsPassword 매개 변수의 값을 두 번 클릭하고 **true**를 선택합니다. 이렇게 하면 텍스트 필드에 입력한 값이 별표로 가려집니다.

5 [구성 요소] 패널의 TextInput 구성 요소를 스테이지로 드래그하고 다음 값을 지정합니다.

- 인스턴스 이름으로 **confirmTi**를 입력합니다.
- W 값으로 **150**을 입력합니다.
- X 값으로 **190**을 입력합니다.
- Y 값으로 **200**을 입력합니다.
- [매개 변수] 섹션에서 displayAsPassword 매개 변수의 값을 두 번 클릭하고 **true**를 선택합니다. 이렇게 하면 텍스트 필드에 입력한 값이 별표로 가려집니다.

6 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
function tiListener(evt_obj:Event){
if(confirmTi.text != pwdTi.text || confirmTi.length < 8)
{
        trace("Password is incorrect. Please reenter it.");
}
else {
        trace("Your password is: " + confirmTi.text);
}
}
confirmTi.addEventListener("enter", tiListener);
```

이 코드는 confirmTi라는 TextInput 인스턴스에 enter 이벤트 핸들러를 설정합니다. 두 매개 변수가 일치하지 않거나 사용자가 8자 미만의 문자를 입력하면 "Password is incorrect. Please reenter it."이라는 메시지가 표시됩니다. 암호가 8자 이상이고 서로 일치하면 [출력] 패널에 입력한 값이 표시됩니다.

7 [컨트롤] > [무비 테스트]를 선택합니다.

ActionScript를 사용하여 TextInput 인스턴스 만들기

다음 예제에서는 ActionScript를 사용하여 TextInput 구성 요소를 만듭니다. 또한 사용자에게 이름을 입력하라고 알리는 데 사용할 프롬프트 Label을 만들고 대문자와 소문자, 마침표 및 공백만 입력할 수 있도록 구성 요소의 restrict 속성을 설정합니다. Label 구성 요소와 TextInput 구성 요소에 텍스트 서식을 지정하는 데 사용되는 TextFormat 객체도 만듭니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 TextInput 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [구성 요소] 패널의 Label 구성 요소를 [라이브러리] 패널로 드래그합니다.

4 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.Label;
import fl.controls.TextInput;

var nameLabel:Label = new Label();
var nameTi:TextInput = new TextInput();
var tf:TextFormat = new TextFormat();

addChild(nameLabel);
addChild(nameTi);

nameTi.restrict = "A-Z .a-z";

tf.font = "Georgia";
tf.color = 0x0000CC;
tf.size = 16;

nameLabel.text = "Name: " ;
nameLabel.setSize(50, 25);
nameLabel.move(100,100);
nameLabel.setStyle("textFormat", tf);
nameTi.move(160, 100);
nameTi.setSize(200, 25);
nameTi.setStyle("textFormat", tf);
```

5 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

TileList 구성 요소 사용

TileList 구성 요소는 데이터 공급자가 데이터와 함께 제공하는 행과 열의 목록으로 구성됩니다. 항목은 TileList의 한 셀에 저장되어 있는 데이터 단위를 나타냅니다. 원래 데이터 공급자가 제공하는 항목에는 일반적으로 label 속성과 source 속성이 있습니다. label 속성은 셀에 표시할 내용을 식별하며 source 속성은 해당 값을 제공합니다.

Array 인스턴스를 만들거나 서버에서 검색할 수 있습니다. TileList 구성 요소에는 해당 데이터 공급자로 프록시되는 메서드(예: addItem() 및 removeItem())가 있습니다. 목록에 외부 데이터 공급자가 제공되지 않으면 이러한 메서드가 자동으로 데이터 공급자 인스턴스를 만듭니다. 이 인스턴스는 List.dataProvider를 통해 표시됩니다.

TileList 구성 요소와 사용자의 상호 작용

TileList의 각 셀은 ICellRenderer 인터페이스를 구현하는 Sprite를 사용하여 렌더링됩니다. TileList cellRenderer 속성을 사용하면 이 렌더러를 지정할 수 있습니다. TileList 구성 요소의 기본 CellRenderer는 이미지(클래스, 비트맵, 인스턴스 또는 URL)를 표시하는 ImageCell이며, 선택적으로 레이블을 사용할 수 있습니다. 레이블은 항상 셀 아래쪽에 정렬되는 한 행으로 된 텍스트입니다. TileList는 한 방향으로만 스크롤할 수 있습니다.

TileList 인스턴스에 포커스가 있으면 다음 키를 사용하여 내부 항목에 액세스할 수 있습니다.

키	설명
위쪽 화살표 및 아래쪽 화살표	위 또는 아래 열로 이동할 수 있습니다. allowMultipleSelection 속성이 true이면 이 키와 Shift 키를 조합하여 여러 셀을 선택할 수 있습니다.
왼쪽 화살표 및 오른쪽 화살표	왼쪽 또는 오른쪽 행으로 이동할 수 있습니다. allowMultipleSelection 속성이 true이면 이 키와 Shift 키를 조합하여 여러 셀을 선택할 수 있습니다.
홈	TileList에서 첫 번째 셀을 선택합니다. allowMultipleSelection 속성이 true인 경우, Shift 키를 누른 상태에서 Home 키를 누르면 현재 선택한 셀부터 첫 번째 셀까지 모두 선택됩니다.
End	TileList에서 마지막 셀을 선택합니다. allowMultipleSelection 속성이 true인 경우, Shift 키를 누른 상태에서 End 키를 누르면 현재 선택한 셀부터 마지막 셀까지 모두 선택됩니다.
Ctrl	allowMultipleSelection 속성이 true인 경우 임의의 순서로 여러 셀을 선택할 수 있습니다.

응용 프로그램에 TileList 구성 요소를 추가할 때 다음 ActionScript 코드를 추가하여 화면 판독기에서 해당 구성 요소에 액세스할 수 있게 만들 수 있습니다.

```
import fl.accessibility.TileListAccImpl;

TileListAccImpl.enableAccessibility();
```

구성 요소의 인스턴스 수에 관계없이 구성 요소의 액세스 가능성을 한 번만 활성화하면 됩니다. 자세한 내용은 **Flash** 사용 설명서에서 18장, "액세스 가능한 내용 만들기"를 참조하십시오.

TileList 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 TileList 구성 요소 인스턴스에 대해 allowMultipleSelection, columnCount, columnWidth, dataProvider, direction, horizontalScrollLineSize, horizontalScrollPageSize, labels, rowCount, rowHeight, ScrollPolicy, verticalScrollLineSize, verticalScrollPageSize 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. dataProvider 매개 변수 사용에 대한 자세한 내용은 26페이지의 “dataProvider 매개 변수 사용”을 참조하십시오.

메서드, 속성 및 이벤트를 사용하여 TileList 인스턴스에 대한 추가 옵션을 설정하도록 ActionScript를 작성할 수 있습니다. 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 TileList 클래스를 참조하십시오.

TileList 구성 요소를 사용하여 응용 프로그램 만들기

다음 예제에서는 MovieClip을 사용하여 그리기 색상의 배열로 TileList를 채웁니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. TileList 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aTl**로 지정합니다.
3. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.data.DataProvider;
import flash.display.DisplayObject;

var aBoxes:Array = new Array();
var i:uint = 0;
var colors:Array = new Array(0x00000, 0xFF0000, 0x0000CC, 0x00CC00, 0xFFFF00);
var colorNames:Array = new Array("Midnight", "Cranberry", "Sky", "Forest", "July");
var dp:DataProvider = new DataProvider();
for(i=0; i < colors.length; i++) {
    aBoxes[i] = new MovieClip();
    drawBox(aBoxes[i], colors[i]);// draw box w next color in array
    dp.addItem( {label:colorNames[i], source:aBoxes[i]} );
}
aTl.dataProvider = dp;
aTl.columnWidth = 110;
aTl.rowHeight = 130;
aTl.setSize(280,150);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(0, 0, 100, 100);
            box.graphics.endFill();
}
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 테스트합니다.

ActionScript를 사용하여 TileList 구성 요소 만들기

다음 예제에서는 TileList 인스턴스를 동적으로 만들고 ColorPicker, ComboBox, NumericStepper 및 CheckBox 구성 요소의 인스턴스를 해당 인스턴스에 추가합니다. 그런 다음 표시할 구성 요소의 레이블과 이름을 포함하는 Array를 만들고 해당 Array(dp)를 TileList의 dataProvider 속성에 지정합니다. 또한 columnWidth 및 rowHeight 속성과 setSize() 메서드를 사용하여 TileList를 배치하고, move() 메서드를 사용하여 스테이지에 TileList를 배치한 다음, contentPadding 스타일을 사용하여 TileList 인스턴스의 테두리와 해당 내용 사이에 공백을 만들고, sortItemsOn() 메서드를 사용하여 레이블 기준으로 내용을 정렬합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 ColorPicker, ComboBox, NumericStepper, CheckBox 및 TileList 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.controls.CheckBox;
import fl.controls.ColorPicker;
import fl.controls.ComboBox;
import fl.controls.NumericStepper;
import fl.controls.TileList;
import fl.data.DataProvider;

var aCp:ColorPicker = new ColorPicker();
var aCb:ComboBox = new ComboBox();
var aNs:NumericStepper = new NumericStepper();
var aCh:CheckBox = new CheckBox();
var aTl:TileList = new TileList();

var dp:Array = [
{label:"ColorPicker", source:aCp},
{label:"ComboBox", source:aCb},
{label:"NumericStepper", source:aNs},
{label:"CheckBox", source:aCh},
];
aTl.dataProvider = new DataProvider(dp);
aTl.columnWidth = 110;
aTl.rowHeight = 100;
aTl.setSize(280,130);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);
aTl.sortItemsOn("label");
addChild(aTl);
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 테스트합니다.

UILoader 구성 요소 사용

UILoader 구성 요소는 SWF, JPEG, 점진적 JPEG, PNG 및 GIF 파일을 표시할 수 있는 컨테이너입니다. 원격 위치에서 내용을 가져와 Flash 응용 프로그램에 보내야 할 때마다 UILoader를 사용할 수 있습니다. 예를 들어, UILoader를 사용하여 회사 로고(JPEG 파일)를 양식에 추가할 수 있습니다. 또한 사진을 표시하는 응용 프로그램에서 UILoader 구성 요소를 사용할 수 있습니다. load() 메서드를 사용하여 내용을 로드하고, percentLoaded 속성을 사용하여 로드된 내용의 양을 확인하고, complete 이벤트를 사용하여 로드가 완료된 시점을 확인할 수 있습니다.

UILoader의 내용 크기를 조절하거나 UILoader 자체의 크기를 조절하여 내용의 크기에 맞출 수 있습니다. 기본적으로 내용의 크기는 UILoader에 맞게 조절됩니다. 런타임에 내용을 로드할 수 있고 로드 진행률을 모니터링할 수도 있습니다. 그러나 내용을 한 번 로드하면 해당 내용이 캐시되기 때문에 진행률이 신속하게 100%에 도달합니다. UILoader에 내용을 로드할 때 위치를 지정하려면 0, 0(X 및 Y 좌표)과 같은 형태로 위치를 지정해야 합니다.

UILoader 구성 요소와 사용자의 상호 작용

UILoader 구성 요소는 포커스를 받을 수 없습니다. 그러나 UILoader 구성 요소에 로드된 내용은 포커스를 받을 수 있으며 해당 포커스와 상호 작용합니다. 포커스 제어에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0** 참조 설명서의 FocusManager 클래스 및 24페이지의 “FocusManager를 사용한 작업”을 참조하십시오.

UILoader 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 UILoader 구성 요소 인스턴스에 대해 autoLoad, maintainAspectRatio, source, scaleContent 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다.

각 UILoader 인스턴스의 실시간 미리 보기에는 제작하는 동안 속성 관리자나 [구성 요소 관리자]에서 변경한 매개 변수가 반영됩니다.

메서드, 속성 및 이벤트를 사용하여 UILoader 인스턴스에 대한 추가 옵션을 설정하도록 ActionScript를 작성할 수 있습니다. 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 UILoader 클래스를 참조하십시오.

UILoader 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램을 제작하는 동안 UILoader 구성 요소를 추가하는 방법을 설명합니다. 이 예제에서 로더는 로고 GIF 이미지를 로드합니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널의 UILoader 구성 요소를 스테이지로 드래그합니다.
3. 속성 관리자에서 인스턴스 이름으로 **aUI**를 입력합니다.
4. 스테이지에서 로더를 선택하고 [구성 요소 관리자]에서 source 매개 변수 값으로 **http://www.helpexamples.com/images/logo.gif**를 입력합니다.

ActionScript를 사용하여 UILoader 구성 요소 인스턴스 만들기

다음 예제에서는 ActionScript를 사용하여 UILoader 구성 요소를 만들고 꽃 이미지(JPEG)를 로드합니다. 또한 complete 이벤트가 발생하면 [출력] 패널에 로드된 바이트 수를 표시합니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널의 UILoader 구성 요소를 [라이브러리] 패널로 드래그합니다.
3. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import fl.containers.UILoader;

var aLoader:UILoader = new UILoader();
aLoader.source = "http://www.flash-mx.com/images/image1.jpg";
aLoader.scaleContent = false;
addChild(aLoader);

aLoader.addEventListener(Event.COMPLETE, completeHandler);
function completeHandler(event:Event) {
    trace("Number of bytes loaded: " + aLoader.bytesLoaded);
}
```

4. [컨트롤] > [무비 테스트]를 선택합니다.

UIScrollBar 구성 요소 사용

UIScrollBar 구성 요소를 사용하면 텍스트 필드에 스크롤 막대를 추가할 수 있습니다. ActionScript를 사용하여 제작하는 동안이나 런타임에 텍스트 필드에 스크롤 막대를 추가할 수 있습니다. UIScrollBar 구성 요소를 사용하려면 스테이지에 텍스트 필드를 만들고 [구성 요소] 패널에서 텍스트 필드 경계 상자의 사면 중 하나로 UIScrollBar 구성 요소를 드래그합니다.

스크롤 막대의 길이가 스크롤 화살표를 합친 크기보다 작으면 올바르게 표시되지 않고 화살표 버튼 중 하나가 다른 화살표 뒤에 숨겨집니다. Flash에서는 이와 관련된 오류 검사를 하지 않습니다. 이 경우 ActionScript를 사용하여 스크롤 막대를 숨기는 것이 좋습니다. 스크롤 막대의 크기를 조절한 결과 스크롤 상자(축소판)를 표시할 공간이 부족하게 되면 스크롤 상자가 표시되지 않습니다.

UIScrollBar 구성 요소는 다른 스크롤 막대와 비슷한 기능을 하며 양쪽 끝에는 화살표 버튼이, 그 사이에는 스크롤 트랙과 스크롤 상자(축소판)가 있습니다. 이 구성 요소를 텍스트 필드의 가장자리에 첨부하고 가로와 세로 양쪽 방향으로 사용할 수도 있습니다.

TextField에 대한 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 TextField 클래스를 참조하십시오.

UIScrollBar 구성 요소와 사용자의 상호 작용

다른 여러 구성 요소와 달리 UIScrollBar 구성 요소에는 클릭을 반복할 필요 없이 마우스 버튼을 누른 채로 있는 것과 같은 연속된 마우스 입력을 사용할 수 있습니다.

UIScrollBar 구성 요소와 키보드는 상호 작용하지 않습니다.

UIScrollBar 구성 요소 매개 변수

속성 관리자나 구성 요소 관리자에서 각 UIScrollBar 구성 요소 인스턴스에 대해 direction, scrollTargetName 등의 제작 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다.

클래스 메서드, 속성 및 이벤트를 사용하여 UIScrollBar 인스턴스에 대한 추가 옵션을 설정하도록 ActionScript를 작성할 수 있습니다. 자세한 내용은 **ActionScript 3.0 Reference for Flash Professional**에서 UIScrollBar 클래스를 참조하십시오.

UIScrollBar 구성 요소를 사용하여 응용 프로그램 만들기

다음 절차에서는 응용 프로그램을 제작하는 동안 UIScrollBar 구성 요소를 추가하는 방법을 설명합니다.

1. 새 Flash(ActionScript 3.0) 문서를 만듭니다.
2. 한두 행의 텍스트를 입력할 수 있는 높이의 동적 텍스트 필드를 만들고 속성 관리자에서 인스턴스 이름으로 **myText**를 지정합니다.
3. 스크롤 막대를 가로로 사용하려는 경우 속성 관리자에서 입력 텍스트 필드의 [행 유형]을 [여러 행] 또는 [여러 행을 한행으로]로 설정합니다.
4. [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다. 이 코드는 사용자가 스크롤해야만 모든 내용을 볼 수 있도록 text 속성을 채웁니다.

   ```
   myText.text="When the moon is in the seventh house and Jupiter aligns with Mars, then peace will guide the planet and love will rule the stars."
   ```

   참고: 스테이지에서 모든 텍스트를 보려면 스크롤해야 할 만큼 텍스트 필드가 작은지 확인합니다. 텍스트 필드가 충분히 작지 않으면 스크롤 막대가 나타나지 않거나 스크롤 상자(내용을 스크롤할 때 드래그하는 부분)없이 두 줄로 나타날 수 있습니다.
5. [보기] > [물리기] > [객체에 물리기]를 선택하여 객체 물리기를 설정했는지 확인합니다.
6. [구성 요소] 패널에서 인스턴스를 첨부하려는 위치에 가까운 텍스트 입력 필드로 UIScrollBar 인스턴스를 드래그합니다. 구성 요소가 제대로 바인딩되려면 마우스를 놓을 때 구성 요소와 텍스트 필드가 겹쳐야 합니다. 인스턴스 이름을 **mySb**로 지정합니다.

   구성 요소의 scrollTargetName 속성은 자동으로 속성 관리자와 [구성 요소 관리자]에 있는 텍스트 필드 인스턴스 이름으로 채워집니다. [매개 변수] 탭에 텍스트 필드 인스턴스 이름이 나타나지 않는 경우 UIScrollBar 인스턴스가 충분히 겹쳐지지 않았기 때문일 수 있습니다.
7. [컨트롤] > [무비 테스트]를 선택합니다.

ActionScript를 사용하여 UIScrollBar 구성 요소 인스턴스 만들기

ActionScript로 UIScrollBar 인스턴스를 만들어 런타임에 텍스트 필드와 연결할 수 있습니다. 다음 예제에서는 가로 방향의 UIScrollBar 인스턴스를 만들고 URL로부터 텍스트를 로드하는 **myTxt**라는 텍스트 필드 인스턴스 아래쪽에 연결합니다. 또한 스크롤 막대의 크기를 텍스트 필드의 크기와 일치하도록 설정합니다.

1 새 Flash(ActionScript 3.0) 문서를 만듭니다.

2 ScrollBar 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 ActionScript 코드를 입력합니다.

```
import flash.net.URLLoader;
import fl.controls.UIScrollBar;
import flash.events.Event;

var myTxt:TextField = new TextField();
myTxt.border = true;
myTxt.width = 200;
myTxt.height = 16;
myTxt.x = 200;
myTxt.y = 150;

var mySb:UIScrollBar = new UIScrollBar();
mySb.direction = "horizontal";
// Size it to match the text field.
mySb.setSize(myTxt.width, myTxt.height);

// Move it immediately below the text field.
mySb.move(myTxt.x, myTxt.height + myTxt.y);

// put them on the Stage
addChild(myTxt);
addChild(mySb);
// load text
var loader:URLLoader = new URLLoader();
var request:URLRequest = new URLRequest("http://www.helpexamples.com/flash/lorem.txt");
loader.load(request);
loader.addEventListener(Event.COMPLETE, loadcomplete);

function loadcomplete(event:Event) {
    // move loaded text to text field
    myTxt.text = loader.data;
    // Set myTxt as target for scroll bar.
    mySb.scrollTarget = myTxt;
}
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

5장: UI 구성 요소 사용자 정의

UI 구성 요소 사용자 정의

다음 두 요소 중 하나 또는 둘 모두를 수정하여 응용 프로그램의 구성 요소 모양을 사용자 정의할 수 있습니다.

스타일 각 구성 요소의 스타일 집합을 설정하여 Flash에서 구성 요소 모양을 렌더링하는 데 사용되는 값을 지정할 수 있습니다. 일반적으로 스타일에 따라 각 상태의 구성 요소에 사용할 스킨과 아이콘이 결정되며 사용할 텍스트 서식과 패딩 값도 결정됩니다.

스킨 스킨은 특정 상태의 구성 요소 그래픽 모양을 구성하는 심볼 모음으로 이루어져 있습니다. 사용할 스킨은 스타일에 따라 결정되지만 Flash가 구성 요소를 그릴 때 사용하는 그래픽 요소는 스킨입니다. **스키닝**은 구성 요소의 그래픽을 수정하거나 바꾸어 구성 요소의 모양을 변경하는 과정입니다.

참고: ActionScript 3.0 구성 요소의 기본 모양은 일종의 테마(Aeon Halo)로 생각할 수 있지만 이러한 스킨은 구성 요소에 내장되어 있습니다. ActionScript 2.0 구성 요소와 달리 ActionScript 3.0 구성 요소는 외부 테마 파일을 지원하지 않습니다.

스타일 설정

Flash에서 다양한 상태의 구성 요소를 그릴 때 사용되는 스킨, 아이콘, 텍스트 서식 및 패딩은 일반적으로 구성 요소의 스타일에 따라 결정됩니다. 예를 들어, Flash에서는 마우스 버튼으로 Button을 클릭할 때 발생하는 다운 상태를 업 또는 보통 상태와는 다르게 표시하기 위해 다른 스킨을 사용하여 Button을 그립니다. 또한 Button의 enabled 속성을 false로 설정하여 버튼이 비활성 상태가 된 경우에도 다른 스킨을 사용합니다.

문서, 클래스 및 인스턴스 수준에서 구성 요소의 스타일을 설정할 수 있습니다. 또한 일부 스타일 속성은 부모 구성 요소로부터 상속받을 수 있습니다. 예를 들어, List 구성 요소는 BaseScrollPane에서 상속되는 ScrollBar 스타일을 상속받습니다.

다음과 같은 방식으로 스타일을 설정하여 구성 요소를 사용자 정의할 수 있습니다.

- 구성 요소 인스턴스에 스타일을 설정합니다. 단일 구성 요소 인스턴스의 색상과 텍스트 속성을 변경할 수 있습니다. 특정 상황에서는 이 방법이 효율적일 수 있지만 문서 내에 있는 모든 구성 요소에 대해 속성을 개별적으로 설정하려면 너무 많은 시간이 소요될 수 있습니다.
- 문서에서 특정 유형의 모든 구성 요소에 스타일을 설정합니다. 특정 유형의 모든 구성 요소(예: 문서의 모든 CheckBox 또는 모든 Button)에 일관된 모양을 적용하려면 구성 요소 수준에서 스타일을 설정합니다.

  컨테이너에 설정된 스타일 속성 값은 포함된 구성 요소에 상속됩니다.

  Flash에서는 실시간 미리 보기 기능을 사용하여 스테이지에서 구성 요소를 볼 때 스타일 속성의 변경 내용이 표시되지 않습니다.

스타일 설정 이해

다음은 스타일 사용에 대한 몇 가지 주요 사항입니다.

상속 자식 구성 요소는 부모 구성 요소로부터 스타일을 상속받도록 설정되어 있습니다. ActionScript 내에서는 스타일 상속을 설정할 수 없습니다.

우선 순위 구성 요소 스타일이 여러 가지 방법으로 설정된 경우 우선 순위에 따라 첫 번째 스타일이 사용됩니다. Flash에서는 값을 발견할 때까지 다음 순서로 스타일을 찾습니다.

1 구성 요소 인스턴스에서 스타일 속성을 찾습니다.

2 스타일이 상속되는 스타일에 속하면 부모 계층에서 상속된 값을 찾습니다.

3 구성 요소에서 스타일을 찾습니다.

4 StyleManager에서 전역 설정을 찾습니다.

5 아직 속성이 정의되지 않은 경우 속성의 값은 undefined입니다.

구성 요소의 기본 스타일 액세스

구성 요소 클래스의 정적 getDefaultStyles() 메서드를 사용하여 구성 요소의 기본 스타일에 액세스할 수 있습니다. 예를 들어, 다음 코드는 ComboBox 구성 요소의 기본 스타일을 검색하여 buttonWidth 및 downArrowDownSkin 속성의 기본값을 표시합니다.

```
import fl.controls.ComboBox;
var styleObj:Object = ComboBox.getStyleDefinition();
trace(styleObj.buttonWidth); // 24
trace(styleObj.downArrowDownSkin); // ScrollArrowDown_downSkin
```

구성 요소 인스턴스의 스타일 설정 및 가져오기

모든 UI 구성 요소 인스턴스는 setStyle() 및 getStyle() 메서드를 호출하여 직접 스타일을 설정하거나 검색할 수 있습니다. 다음 구문은 구성 요소 인스턴스의 스타일과 값을 설정합니다.

```
instanceName.setStyle("styleName", value);
```

이 구문은 구성 요소 인스턴스의 스타일을 검색합니다.

```
var a_style:Object = new Object();
a_style = instanceName.getStyle("styleName");
```

getStyle() 메서드는 데이터 유형이 각기 다른 여러 스타일을 반환할 수 있으므로 Object 유형을 반환합니다. 예를 들어, 다음 코드는 TextArea 인스턴스(aTa)의 글꼴 스타일을 설정한 다음 getStyle() 메서드를 사용하여 글꼴 스타일을 검색합니다. 이 예제에서는 반환값을 TextFormat 객체로 변환하여 TextFormat 변수에 지정합니다. 형 변환을 하지 않을 경우 Object 변수를 TextFormat 변수로 강제 형 변환하는 과정에서 컴파일러 오류가 발생합니다.

```
import flash.text.TextFormat;

var tf:TextFormat = new TextFormat();
tf.font = "Georgia";
aTa.setStyle("textFormat",tf);
aTa.text = "Hello World!";
var aStyle:TextFormat = aTa.getStyle("textFormat") as TextFormat;
trace(aStyle.font);
```

TextFormat을 사용하여 텍스트 속성 설정

TextFormat 객체를 사용하여 구성 요소 인스턴스의 텍스트 서식을 지정할 수 있습니다. TextFormat 객체에는 bold, bullet, color, font, italic, size 등의 텍스트 특성을 지정할 수 있는 속성이 있습니다. TextFormat 객체에서 이러한 속성을 설정한 다음 setStyle() 메서드를 호출하여 구성 요소 인스턴스에 적용할 수 있습니다. 예를 들어, 다음 코드는 TextFormat 객체의 font, size 및 bold 속성을 설정하여 Button 인스턴스에 적용합니다.

```
/* Create a new TextFormat object to set text formatting properties. */
var tf:TextFormat = new TextFormat();
tf.font = "Arial";
tf.size = 16;
tf.bold = true;
a_button.setStyle("textFormat", tf);
```

다음은 Submit 레이블이 지정된 버튼에 이러한 설정을 적용한 결과를 나타낸 그림입니다.

setStyle()을 통해 구성 요소 인스턴스에 설정한 스타일 속성은 우선 순위가 가장 높기 때문에 다른 모든 스타일 설정은 무시됩니다. 그러나 단일 구성 요소 인스턴스에 setStyle()을 사용하여 설정한 속성이 많을수록 런타임에 구성 요소의 렌더링 속도가 느려집니다.

구성 요소의 모든 인스턴스에 하나의 스타일 설정

StyleManager 클래스의 정적 setComponentStyle() 메서드를 사용하면 구성 요소 클래스의 모든 인스턴스에 하나의 스타일을 설정할 수 있습니다. 예를 들어, Button을 스테이지로 드래그한 후 타임라인의 프레임 1에 있는 [액션] 패널에 다음 ActionScript 코드를 추가하여 모든 Button의 텍스트 색상을 빨강으로 설정할 수 있습니다.

```
import fl.managers.StyleManager;
import fl.controls.Button;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setComponentStyle(Button, "textFormat", tf);
```

이후 스테이지에 추가하는 모든 Button은 빨강 레이블을 가지게 됩니다.

모든 구성 요소에 하나의 스타일 설정

StyleManager 클래스의 정적 setStyle() 메서드를 사용하여 모든 구성 요소에 하나의 스타일을 설정할 수 있습니다.

1. List 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aList**로 지정합니다.
2. Button 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aButton**으로 지정합니다.
3. [액션] 패널이 열려 있지 않은 경우 **F9** 키를 누르거나 [윈도우] 메뉴에서 [액션]을 선택하여 [액션] 패널을 열고 타임라인의 프레임 1에 다음 코드를 입력하여 모든 구성 요소의 텍스트 색상을 빨강으로 설정합니다.

   ```
   import fl.managers.StyleManager;

   var tf:TextFormat = new TextFormat();
   tf.color = 0xFF0000;
   StyleManager.setStyle("textFormat", tf);
   ```

4. [액션] 패널에 다음 코드를 추가하여 List를 텍스트로 채웁니다.

   ```
   aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
   aList.addItem({label:"1966 Mustang (Classic)", data:27000});
   aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
   aList.allowMultipleSelection = true;
   ```

5. [컨트롤] > [무비 테스트]를 선택하거나 Ctrl+Enter를 눌러 코드를 컴파일하고 내용을 테스트합니다. 버튼 레이블과 목록의 텍스트가 모두 빨간 색이어야 합니다.

스킨

구성 요소의 모양은 외곽선, 채움 색상, 아이콘 등과 같은 그래픽 요소로 구성되며 심지어 다른 구성 요소로 구성되는 경우도 있습니다. 예를 들어, ComboBox는 List 구성 요소를 포함하고 List 구성 요소는 ScrollBar를 포함합니다. 이러한 그래픽 요소가 모여 ComboBox의 모양을 만듭니다. 그러나 구성 요소의 모양은 구성 요소의 현재 상태에 따라 변합니다. 예를 들어, 레이블이 없는 CheckBox는 응용 프로그램에서 다음과 비슷한 모양으로 표시됩니다.

일반적인 업 상태의 CheckBox

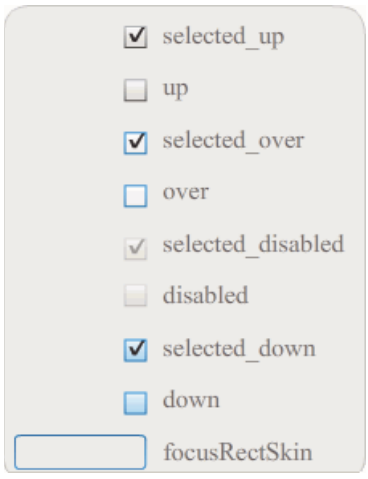
CheckBox 위에서 마우스 버튼을 클릭한 채로 있으면 모양이 다음과 같이 변합니다.

다운 상태의 CheckBox

마우스 버튼을 놓으면 CheckBox가 원래 모양으로 돌아가지만 선택되었음을 나타내는 체크 표시가 나타납니다.

선택 상태의 CheckBox

다양한 상태의 구성 요소를 나타내는 아이콘을 통칭하여 스킨이라고 합니다. Flash의 다른 심볼과 마찬가지로 어느 상태의 구성 요소든 Flash에서 해당 스킨을 편집하여 모양을 변경할 수 있습니다. 구성 요소 스킨은 두 가지 방법으로 액세스할 수 있습니다. 가장 쉬운 방법은 구성 요소를 스테이지로 드래그하고 두 번 클릭하는 것입니다. 그러면 구성 요소 스킨의 팔레트가 열립니다. 다음은 CheckBox의 스킨 팔레트입니다.



CheckBox의 스킨

[라이브러리] 패널에서 개별적으로 구성 요소 스킨에 액세스할 수도 있습니다. 스테이지로 구성 요소를 드래그하면 구성 요소와 함께 구성 요소의 에셋 폴더와 구성 요소에 포함된 다른 구성 요소가 라이브러리에 복사됩니다. 예를 들어, ComboBox를 스테이지로 드래그하면 [라이브러리] 패널에는 ComboBox를 구성하는 List, ScrollBar 및 TextInput 구성 요소를 비롯하여 이러한 구성 요소 각각의 스킨 폴더와 이러한 구성 요소의 공유 요소가 들어 있는 Shared Assets 폴더가 포함됩니다. 이러한 구성 요소의 스킨을 편집하려면 해당 스킨 폴더(ComboBoxSkins, ListSkins, ScrollBarSkins 또는 TextInputSkins)를 열고 편집하려는 스킨의 아이콘을 두 번 클릭합니다. 예를 들어, ComboBox_downSkin을 두 번 클릭하면 다음 그림과 같이 스킨이 심볼 편집 모드로 열립니다.

ComboBox_downSkin

새 스킨 만들기

문서에 있는 구성 요소의 모양을 새로 만들려면 구성 요소의 스킨을 편집하여 모양을 변경합니다. 구성 요소의 스킨에 액세스하려면 스테이지에서 구성 요소를 두 번 클릭하여 해당 스킨 팔레트를 엽니다. 그런 다음 편집하려는 스킨을 두 번 클릭하여 심볼 편집 모드로 엽니다. 예를 들어, 스테이지에서 TextArea 구성 요소를 두 번 클릭하여 에셋을 심볼 편집 모드로 엽니다. 그리고 나서 확대/축소 컨트롤을 400% 또는 원하는 경우 그 이상으로 설정하고 심볼을 편집하여 모양을 변경합니다. 작업을 완료하면 문서에 있는 구성 요소의 모든 인스턴스에 변경 내용이 적용됩니다. 다른 방법은 [라이브러리] 패널에서 특정 스킨을 두 번 클릭하여 스테이지에서 심볼 편집 모드로 여는 것입니다.

다음과 같은 방법으로 구성 요소 스킨을 수정할 수 있습니다.

- 모든 인스턴스에 대해 새 스킨을 만듭니다.
- 일부 인스턴스에 대해 새 스킨을 만듭니다.

모든 인스턴스에 대한 스킨 만들기

구성 요소의 스킨을 편집하면 기본적으로 문서에 있는 해당 구성 요소의 모든 인스턴스 모양이 변경됩니다. 동일한 구성 요소에 여러 가지 서로 다른 모양을 만들려면 변경하려는 스킨을 복제해서 각기 다른 이름을 지정하고 편집한 다음 적절한 스타일을 설정하여 적용해야 합니다. 자세한 내용은 93페이지의 “일부 인스턴스에 대한 스킨 만들기”를 참조하십시오.

이 장에서는 각 UI 구성 요소마다 하나 이상의 스킨을 변경하는 방법에 대해 설명합니다. 이러한 절차 중 하나에 따라 UI 구성 요소의 스킨을 변경하면 문서에서 모든 인스턴스에 대해 스킨이 변경됩니다.

일부 인스턴스에 대한 스킨 만들기

다음과 같은 일반적인 절차에 따라 구성 요소의 일부 인스턴스에 대한 스킨을 만들 수 있습니다.

- [라이브러리] 패널에서 구성 요소의 Assets 폴더에 있는 스킨을 선택합니다.
- 스킨을 복제하고 고유한 클래스 이름을 지정합니다.
- 스킨을 원하는 모양으로 편집합니다.
- 구성 요소 인스턴스의 setStyle() 메서드를 호출하여 스킨 스타일에 새 스킨을 지정합니다.

다음 절차에서는 두 Button 인스턴스 중 하나에 새 selectedDownSkin을 만듭니다.

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널에서 Button 두 개를 스테이지로 드래그하고 인스턴스 이름을 **aButton** 및 **bButton**으로 지정합니다.
3. [라이브러리] 패널을 열고 그 안의 Component Assets 폴더와 ButtonSkins 폴더를 엽니다.
4. selectedDownSkin 스킨을 클릭하여 선택합니다.
5. 마우스 오른쪽 버튼을 클릭하여 컨텍스트 메뉴를 열고 [복제]를 선택합니다.
6. [심볼 복제] 대화 상자에서 새 스킨에 고유한 이름(예: **Button_mySelectedDownSkin**)을 지정합니다. 그런 다음 [확인]을 클릭합니다.
7. [라이브러리] > Component Assets 폴더 > ButtonSkins 폴더에서 Button_mySelectedDownSkin을 선택하고 마우스 오른쪽 버튼을 클릭하여 컨텍스트 메뉴를 엽니다. [링크]를 선택하여 [링크 속성] 대화 상자를 엽니다.

8 [ActionScript에 내보내기] 체크 상자를 클릭합니다. [첫 프레임으로 내보내기] 체크 상자는 선택된 상태로 두고 클래스 이름이 고유한지 확인합니다. [확인]을 클릭하고, 클래스 정의를 찾을 수 없어 새로 만든다는 경고가 나타나면 [확인]을 다시 클릭합니다.

9 [라이브러리] 패널에서 Button_mySelectedDownSkin 스킨을 두 번 클릭하여 심볼 편집 모드로 엽니다.

10 스킨 중앙에 있는 파랑 채움 색상을 클릭하여 속성 관리자의 [채움 색상] 선택기에 해당 색상을 표시합니다. 색상 선택기를 클릭하고 스킨 채움 색상으로 #00CC00을 선택합니다.

11 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

12 속성 관리자에서 각 버튼의 [매개 변수] 탭을 클릭하고 toggle 매개 변수를 true로 설정합니다.

13 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

```
bButton.setStyle("selectedDownSkin", Button_mySelectedDownSkin);
bButton.setStyle("downSkin", Button_mySelectedDownSkin);
```

14 [컨트롤] > [무비 테스트]를 선택합니다.

15 각 버튼을 클릭합니다. bButton 객체의 다운 스킨(selected 및 unselected)에 새 스킨 심볼이 사용되는지 확인합니다.

Button 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 Button 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 Button 클래스 속성을 사용합니다.

버튼의 크기를 조절해도 아이콘이나 레이블의 크기는 변경되지 않습니다. Button의 경계 상자는 Button의 테두리에 해당하며 인스턴스의 히트 영역도 지정합니다. 인스턴스의 크기를 늘리면 히트 영역의 크기도 늘어납니다. 레이블에 비해 경계 상자가 작으면 경계 상자에 맞게 레이블이 잘립니다.

Button에 아이콘이 있고 아이콘이 Button보다 크면 아이콘이 Button 테두리를 넘어갑니다.

Button 구성 요소에 스타일 사용

일반적으로 Button의 스타일에 따라 다양한 상태의 구성 요소를 그릴 때 사용되는 Button의 스킨, 아이콘, 텍스트 서식 및 패딩 값이 결정됩니다.

다음 절차에서는 스테이지에 Button을 두 개 배치하고 사용자가 둘 중 하나를 클릭하면 두 Button의 emphasized 속성이 모두 true로 설정되도록 합니다. 또한 사용자가 두 번째 Button을 클릭하면 emphasizedSkin 스타일이 selectedOverSkin 스타일로 설정되어 동일한 상태의 두 버튼이 다른 스킨으로 표시되도록 합니다.

1 Flash 파일(ActionScript 3.0)을 만듭니다.

2 두 개의 Button을 한 번에 하나씩 스테이지로 드래그하고 **aBtn** 및 **bBtn**이라는 인스턴스 이름을 지정합니다. 속성 관리자의 [매개 변수] 탭에서 Button A 및 Button B라는 레이블을 지정합니다.

3 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

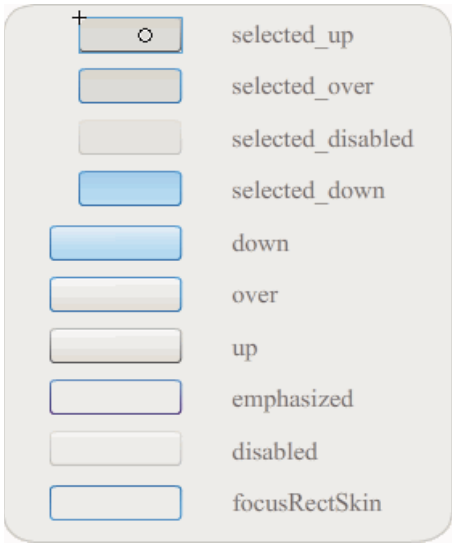
```
bBtn.emphasized = true;
aBtn.emphasized = true;
bBtn.addEventListener(MouseEvent.CLICK, Btn_handler);
function Btn_handler(evt:MouseEvent):void {
    bBtn.setStyle("emphasizedSkin", "Button_selectedOverSkin");
}
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

5 Button 중 하나를 클릭하여 각 Button에 나타나는 emphasizedSkin 스타일의 효과를 확인합니다.

Button 구성 요소에 스킨 사용

Button 구성 요소는 다음과 같이 각기 다른 상태를 나타내는 스킨을 사용합니다. 하나 이상의 스킨을 편집하여 Button의 모양을 변경하려면 스테이지에서 Button 인스턴스를 두 번 클릭하여 다음 그림과 같이 해당 스킨 팔레트를 엽니다.

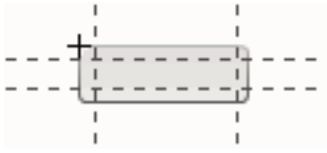


Button 스킨

버튼이 활성화된 경우에는 포인터가 그 위로 이동할 때 버튼이 오버 상태로 표시됩니다. 버튼을 누르면 버튼이 입력 포커스를 받고 다운 상태로 표시됩니다. 마우스를 놓으면 버튼은 오버 상태로 되돌아갑니다. 마우스를 누른 상태에서 포인터를 버튼 밖으로 이동하면 버튼은 원래 상태로 되돌아갑니다. toggle 매개 변수가 true로 설정된 경우, 눌림 상태는 selectedDownSkin, 업 상태는 selectedUpSkin, 오버 상태는 selectedOverSkin으로 표시됩니다.

Button이 비활성화된 경우에는 사용자 상호 작용과 상관없이 비활성 상태로 표시됩니다.

스킨 중 하나를 편집하려면 스킨을 두 번 클릭하여 다음 그림과 같이 심볼 편집 모드로 엽니다.



심볼 편집 모드의 Button

이제 Flash 제작 도구를 사용하여 스킨을 원하는 형태로 편집할 수 있습니다.

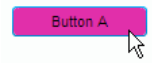
다음 절차에서는 Button의 selected_over 스킨의 색상을 변경합니다.

1. 새 Flash 파일(ActionScript 3.0)을 만듭니다.
2. [구성 요소] 패널의 Button을 스테이지로 드래그합니다. [매개 변수] 탭에서 toggle 매개 변수를 true로 설정합니다.
3. Button을 두 번 클릭하여 해당 스킨 팔레트를 엽니다.
4. selected_over 스킨을 두 번 클릭하여 심볼 편집 모드로 엽니다.
5. 확대/축소 컨트롤을 400%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.
6. 배경을 두 번 클릭하여 속성 관리자의 [채움 색상] 선택기에 배경색을 표시합니다.
7. [채움 색상] 선택기에서 #CC0099 색상을 선택하여 selected_over 스킨의 배경에 적용합니다.
8. 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

9 [컨트롤] > [무비 테스트]를 선택합니다.

10 버튼을 클릭하여 선택 상태로 만듭니다.

마우스 포인터를 Button 위로 이동하면 다음 그림과 같이 selected_over 상태가 나타나야 합니다.



selected_over 스킨의 색상이 변경된 Button

CheckBox 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 CheckBox 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 해당 CheckBox 클래스 속성을 사용합니다. 예를 들어, CheckBox의 크기를 변경하려면 해당하는 height, width, scaleX 및 scaleY 속성을 설정합니다. CheckBox의 크기를 조절해도 레이블이나 체크 상자 아이콘의 크기는 변경되지 않고 경계 상자의 크기만 변경됩니다.

CheckBox 인스턴스의 경계 상자는 표시되지 않으며 다만 인스턴스에 대한 히트 영역을 지정합니다. 인스턴스의 크기를 늘리면 히트 영역의 크기도 늘어납니다. 레이블에 비해 경계 상자가 작으면 경계 상자에 맞게 레이블이 잘립니다.

CheckBox에 스타일 사용

스타일 속성을 설정하여 CheckBox 인스턴스의 모양을 변경할 수 있습니다. 예를 들어, 다음 절차에서는 CheckBox 레이블의 크기와 색상을 변경합니다.

1 [구성 요소] 패널의 CheckBox 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **myCb**로 지정합니다.

2 속성 관리자에서 [매개 변수] 탭을 클릭하고 레이블 매개 변수 값으로 **Less than $500?**를 입력합니다.

3 기본 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 입력합니다.

```
var myTf:TextFormat = new TextFormat();
myCb.setSize(150, 22);
myTf.size = 16;
myTf.color = 0xFF0000;
myCb.setStyle("textFormat", myTf);
```

자세한 내용은 89페이지의 “스타일 설정”을 참조하십시오. 스타일 속성을 설정하여 구성 요소의 아이콘 및 스킨을 변경하는 방법에 대한 자세한 내용은 93페이지의 “새 스킨 만들기” 및 96페이지의 “CheckBox에 스킨 사용”을 참조하십시오.

CheckBox에 스킨 사용

CheckBox 구성 요소에는 다음과 같은 스킨이 있으며, 이러한 스킨을 편집하여 모양을 변경할 수 있습니다.

CheckBox 스킨

이 예제에서는 up 및 selectedUp 상태의 구성 요소 외곽선 색상과 배경색을 변경합니다. 이와 유사한 단계를 따라 다른 상태의 스킨도 변경할 수 있습니다.

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. CheckBox 구성 요소를 스테이지로 드래그합니다. 그러면 라이브러리에도 에셋 폴더와 함께 CheckBox 구성 요소가 배치됩니다.
3. 스테이지에서 CheckBox 구성 요소를 두 번 클릭하여 해당 스킨 아이콘 패널을 엽니다.
4. selected_up 아이콘을 두 번 클릭하여 심볼 편집 모드로 엽니다.
5. 확대/축소 컨트롤을 800%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.
6. CheckBox 테두리를 클릭하여 선택합니다. 속성 관리자의 [채움 색상] 선택기로 #0033FF 색상을 선택하여 테두리에 적용합니다.
7. CheckBox 배경을 두 번 클릭하여 선택하고 [채움 색상] 선택기를 다시 사용하여 배경색을 #00CCFF로 설정합니다.
8. 4~8단계를 반복하여 CheckBox up 스킨을 사용자 정의합니다.
9. [컨트롤] > [무비 테스트]를 선택합니다.

ColorPicker 구성 요소 사용자 정의

ColorPicker의 크기를 조절하는 유일한 방법은 swatchWidth, swatchHeight, backgroundPadding, textFieldWidth 및 textFieldHeight 스타일을 사용하는 것입니다. [변형 도구]를 사용하거나 setSize() 메서드 또는 width, height, scaleX, scaleY 속성을 사용하는 ActionScript로 ColorPicker 크기를 변경하려고 하면 SWF 파일을 만들 때 이러한 값들이 무시되고 ColorPicker가 기본 크기로 표시됩니다. 팔레트 배경은 columnCount 스타일의 setStyle()을 사용하여 설정한 열 수에 맞게 크기가 조절됩니다. 기본 열 수는 18개입니다. 사용자 정의 색상은 1024개까지 설정할 수 있으며 팔레트는 견본 수에 맞게 세로로 크기가 조절됩니다.

ColorPicker 구성 요소에 스타일 사용

다양한 스타일을 설정하여 ColorPicker 구성 요소의 모양을 변경할 수 있습니다. 예를 들어, 다음 절차에서는 ColorPicker의 열 수(columnCount)를 12로 변경하고, 색상 견본의 높이(swatchHeight)와 폭(swatchWidth)을 변경하고, 텍스트 필드(textPadding)와 배경(backgroundPadding)의 패딩을 변경합니다.

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.

2 ColorPicker 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aCp**로 지정합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력합니다.

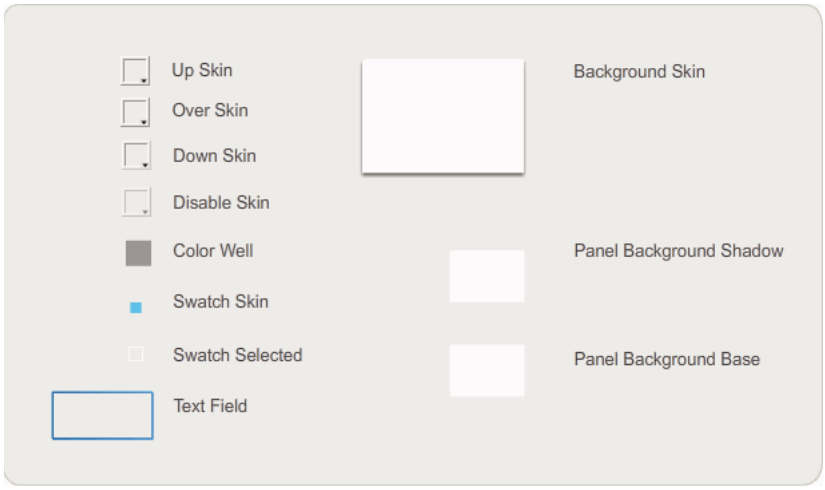
```
aCp.setStyle("columnCount", 12);
aCp.setStyle("swatchWidth", 8);
aCp.setStyle("swatchHeight", 12);
aCp.setStyle("swatchPadding", 2);
aCp.setStyle("backgroundPadding", 3);
aCp.setStyle("textPadding", 7);
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

5 ColorPicker를 클릭하여 열고 이러한 설정이 모양을 어떻게 변경하는지 확인합니다.

ColorPicker 구성 요소에 스킨 사용

ColorPicker 구성 요소는 다음과 같은 스킨을 사용하여 시각적 상태를 표현합니다.



ColorPicker 스킨

배경 스킨의 색상을 변경하여 팔레트 배경색을 변경할 수 있습니다.

1 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.

2 ColorPicker 구성 요소를 스테이지로 드래그합니다.

3 ComboBox 구성 요소를 두 번 클릭하여 해당 스킨 팔레트를 엽니다.

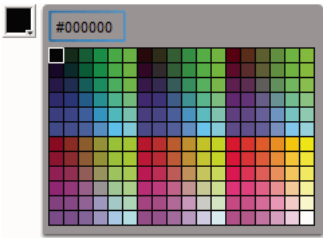
4 배경 스킨을 두 번 클릭하여 선택하고 속성 관리자에 [채움 색상] 선택기를 표시합니다.

5 [채움 색상] 선택기로 #999999 색상을 선택하여 배경 스킨에 적용합니다.

6 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

7 [컨트롤] > [무비 테스트]를 선택합니다.

ColorPicker를 클릭하면 다음 그림과 같이 팔레트 배경이 회색으로 표시되어야 합니다.



배경 스킨이 어두운 회색인 ColorPicker

ComboBox 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 ComboBox 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 ComboBox 클래스 속성을 사용합니다.

ComboBox는 지정한 폭과 높이에 맞게 크기가 조절됩니다. dropdownWidth 속성이 설정되어 있지 않으면 목록 크기가 구성 요소 폭에 맞게 조절됩니다.

ComboBox에 비해 텍스트가 너무 길면 ComboBox에 맞게 텍스트가 잘립니다. ComboBox 크기를 조절하고 텍스트에 맞게 dropdownWidth 속성을 설정해야 합니다.

ComboBox 구성 요소에 스타일 사용

스타일 속성을 설정하여 ComboBox 구성 요소의 모양을 변경할 수 있습니다. 스타일에 따라 구성 요소의 스킨, 셀 렌더러, 패딩 및 버튼 폭 값이 결정됩니다. 다음 예제에서는 buttonWidth 및 textPadding 스타일을 설정합니다. buttonWidth 스타일은 버튼의 히트 영역 폭을 설정하며 ComboBox가 편집 가능할 때 적용됩니다. 이러한 경우 버튼만 누르면 드롭 다운 목록이 열립니다. textPadding 스타일은 텍스트 필드의 바깥쪽 테두리와 텍스트 사이의 간격을 지정합니다. ComboBox를 길게 만드는 경우에는 텍스트 필드에 텍스트를 세로 방향으로 가운데에 오도록 정렬하는 것이 좋습니다. 그렇지 않으면 텍스트 필드의 위쪽에 텍스트가 나타날 수 있습니다.

1 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.

2 ComboBox 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aCb**로 지정합니다.

3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력합니다.

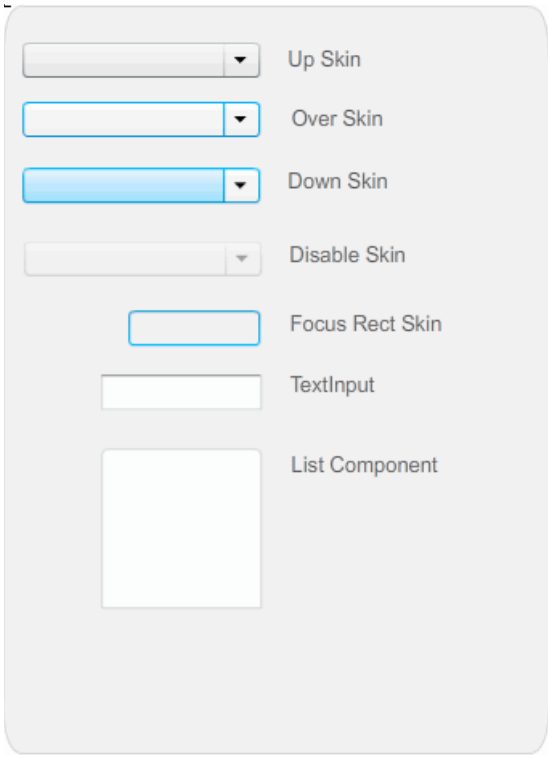
```
import fl.data.DataProvider;

aCb.setSize(150, 35);
aCb.setStyle("textPadding", 10);
aCb.setStyle("buttonWidth", 10);
aCb.editable = true;

var items:Array = [
{label:"San Francisco", data:"601 Townsend St."},
{label:"San Jose", data:"345 Park Ave."},
{label:"San Diego", data:"10590 West Ocean Air Drive, Suite 100"},
{label:"Santa Rosa", data:"2235 Mercury Way, Suite 105"},
{label:"San Luis Obispo", data:"3220 South Higuera Street, Suite 311"}
];
aCb.dataProvider = new DataProvider(items);
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

클릭하여 드롭 다운 목록을 열 수 있는 버튼 영역은 오른쪽에 있는 좁은 영역이며, 텍스트는 텍스트 필드에 세로 방향으로 가운데에 정렬됩니다. 두 setStyle() 문 없이 예제를 실행하여 결과를 확인해 보십시오.

ComboBox에 스킨 사용

ComboBox는 다음과 같은 스킨을 사용하여 시각적 상태를 표현합니다.



ComboBox 스킨

Up 스킨의 색상을 변경하여 스테이지에서 비활성 상태의 구성 요소 색상을 변경할 수 있습니다.

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. ComboBox 구성 요소를 스테이지로 드래그합니다.
3. ComboBox 구성 요소를 두 번 클릭하여 해당 스킨 팔레트를 엽니다.
4. Up 스킨을 두 번 클릭하여 선택하고 편집할 수 있도록 엽니다.
5. 확대/축소 컨트롤을 400%로 설정합니다.
6. 스킨의 중심 영역을 클릭하여 속성 관리자의 [채움 색상] 선택기에 해당 색상을 표시합니다.
7. [채움 색상] 선택기로 #33FF99 색상을 선택하여 Up 스킨에 적용합니다.
8. 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.
9. [컨트롤] > [무비 테스트]를 선택합니다.

   ComboBox가 다음 그림과 같이 스테이지에 나타나야 합니다.

배경 스킨 색상을 사용자 정의한 ComboBox

DataGrid 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 DataGrid 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 width, height, scaleX, scaleY 등의 해당 속성을 사용합니다. 가로 스크롤 막대가 없는 경우 열 폭이 비례에 맞춰 조절됩니다. 열 및 셀 크기가 조절되면 셀의 텍스트가 잘릴 수도 있습니다.

DataGrid 구성 요소에 스타일 사용

스타일 속성을 설정하여 DataGrid 구성 요소의 모양을 변경할 수 있습니다. DataGrid 구성 요소는 List 구성 요소의 스타일을 상속합니다. 자세한 내용은 106페이지의 “List 구성 요소에 스타일 사용”을 참조하십시오.

개별 열 스타일 설정

DataGrid 객체에는 열이 여러 개 포함될 수 있으며 각 열에 서로 다른 셀 렌더러를 지정할 수 있습니다. DataGrid의 각 열은 DataGridColumn 객체로 표현되며 DataGridColumn 클래스에는 열의 CellRenderer를 정의할 수 있는 cellRenderer 속성이 포함됩니다.

1. 새 Flash 문서(ActionScript 3.0)를 만듭니다.
2. DataGrid 구성 요소를 [라이브러리] 패널로 드래그합니다.
3. 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다. 이 코드는 세 번째 열에 긴 텍스트 문자열이 있는 DataGrid를 만듭니다. 마지막으로 열의 cellRenderer 속성을 여러 셀을 렌더링하는 셀 렌더러 이름으로 설정합니다.

```
/* This is a simple cell renderer example.It invokes
the MultiLineCell cell renderer to display a multiple
line text field in one of a DataGrid's columns. */

import fl.controls.DataGrid;
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;
import fl.controls.ScrollPolicy;

// Create a new DataGrid component instance.
var aDg:DataGrid = new DataGrid();

var aLongString:String = "An example of a cell renderer class that displays a multiple line TextField"
var myDP:Array = new Array();
myDP = [{firstName:"Winston", lastName:"Elstad", note:aLongString, item:100},
    {firstName:"Ric", lastName:"Dietrich", note:aLongString, item:101},
    {firstName:"Ewing", lastName:"Canepa", note:aLongString, item:102},
    {firstName:"Kevin", lastName:"Wade", note:aLongString, item:103},
    {firstName:"Kimberly", lastName:"Dietrich", note:aLongString, item:104},
    {firstName:"AJ", lastName:"Bilow", note:aLongString, item:105},
    {firstName:"Chuck", lastName:"Yushan", note:aLongString, item:106},
    {firstName:"John", lastName:"Roo", note:aLongString, item:107},
];

// Assign the data provider to the DataGrid to populate it.
// Note: This has to be done before applying the cellRenderers.
aDg.dataProvider = new DataProvider(myDP);

/* Set some basic grid properties.
```

```
Note: The data grid's row height should reflect
the number of lines you expect to show in the multiline cell.
The cell renderer wil size to the row height.
About 40 for 2 lines or 60 for 3 lines.*/

aDg.columns = ["firstName", "lastName", "note", "item"];
aDg.setSize(430,190);
aDg.move(40,40);
aDg.rowHeight = 40;// Allows for 2 lines of text at default text size.
aDg.columns[0].width = 70;
aDg.columns[1].width = 70;
aDg.columns[2].width = 230;
aDg.columns[3].width = 60;
aDg.resizableColumns = true;
aDg.verticalScrollPolicy = ScrollPolicy.AUTO;
addChild(aDg);
// Assign cellRenderers.
var col3:DataGridColumn = new DataGridColumn();
col3 = aDg.getColumnAt(2);
col3.cellRenderer = MultiLineCell;
```

4 FLA 파일을 MultiLineGrid.fla로 저장합니다.

5 새 ActionScript 파일을 만듭니다.

6 다음 ActionScript 코드를 [스크립트] 윈도우에 복사합니다.

```
package {


    import fl.controls.listClasses.CellRenderer;

    public class MultiLineCell extends CellRenderer
    {

        public function MultiLineCell()
        {
            textField.wordWrap = true;
            textField.autoSize = "left";
        }
        override protected function drawLayout():void {
            textField.width = this.width;
            super.drawLayout();
        }
    }
}
```

7 MultiLineGrid.fla를 저장한 폴더에 ActionScript 파일을 MultiLineCell.as로 저장합니다.

8 MultiLineGrid.fla 응용 프로그램으로 돌아와 [컨트롤] > [무비 테스트]를 선택합니다.

DataGrid는 다음과 비슷해야 합니다.

firstName	lastName	note	item
Winston	Elstad	An example of a cell renderer class that displays a multiple line TextField	100
Ric	Dietrich	An example of a cell renderer class that displays a multiple line TextField	101
Ewing	Canepa	An example of a cell renderer class that displays a multiple line TextField	102
Kevin	Wade	An example of a cell renderer class that displays a multiple line TextField	103

MultiLineGrid.fla 응용 프로그램의 DataGrid

머리글 스타일 설정

headerTextFormat 스타일을 사용하여 머리글 행의 텍스트 스타일을 설정할 수 있습니다. 다음 예제에서는 TextFormat 객체를 사용하여 headerTextFormat 스타일을 Arial 글꼴, 빨강 색상, 글꼴 크기 14 및 기울임체로 설정합니다.

1 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.

2 DataGrid 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aDg**로 지정합니다.

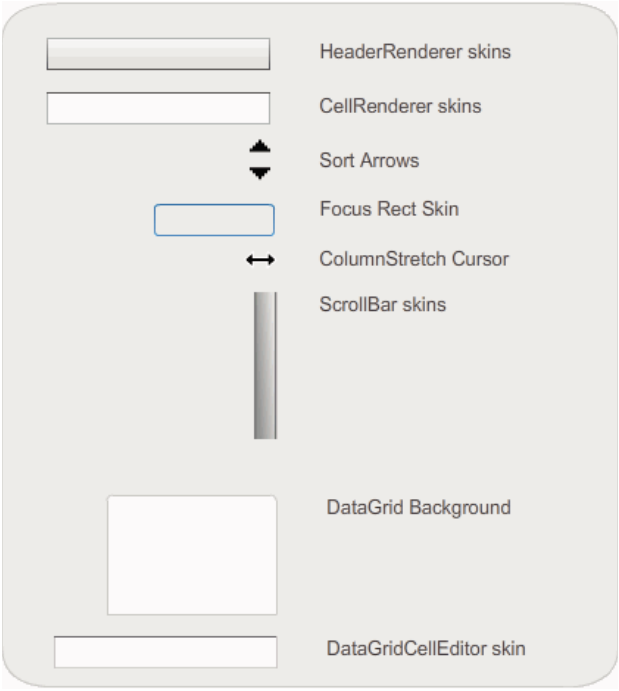
3 [액션] 패널을 열고 기본 타임라인에서 프레임 1을 선택한 후 다음 코드를 입력합니다.

```
import fl.data.DataProvider;
import fl.controls.dataGridClasses.DataGridColumn;

var myDP:Array = new Array();
myDP = [{FirstName:"Winston", LastName:"Elstad"},
    {FirstName:"Ric", LastName:"Dietrich"},
    {FirstName:"Ewing", LastName:"Canepa"},
    {FirstName:"Kevin", LastName:"Wade"},
    {FirstName:"Kimberly", LastName:"Dietrich"},
    {FirstName:"AJ", LastName:"Bilow"},
    {FirstName:"Chuck", LastName:"Yushan"},
    {FirstName:"John", LastName:"Roo"},
];

// Assign the data provider to the DataGrid to populate it.
// Note: This has to be done before applying the cellRenderers.
aDg.dataProvider = new DataProvider(myDP);
aDg.setSize(160,190);
aDg.move(40,40);
aDg.columns[0].width = 80;
aDg.columns[1].width = 80;
var tf:TextFormat = new TextFormat();
tf.size = 14;
tf.color = 0xff0000;
tf.italic = true;
tf.font = "Arial"
aDg.setStyle("headerTextFormat", tf);
```

4 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 실행합니다.

DataGrid 구성 요소에 스킨 사용

DataGrid 구성 요소는 다음과 같은 스킨을 사용하여 시각적 상태를 표현합니다.



DataGrid 스킨

CellRenderer 스킨은 DataGrid의 본문 셀에 사용되며 HeaderRenderer 스킨은 머리글 행에 사용됩니다. 다음 절차에서는 머리글 행의 배경색을 변경하지만 동일한 절차를 사용하여 CellRenderer 스킨을 편집하면 DataGrid 본문 셀의 배경색도 변경할 수 있습니다.

1. 새 Flash 문서(ActionScript 3.0)를 만듭니다.
2. DataGrid 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aDg**로 지정합니다.
3. DataGrid 구성 요소를 두 번 클릭하여 해당 스킨 팔레트를 엽니다.
4. 확대/축소 컨트롤을 400%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.
5. HeaderRenderer 스킨을 두 번 클릭하여 HeaderRenderer 스킨 팔레트를 엽니다.
6. Up_Skin을 두 번 클릭하여 심볼 편집 모드로 엽니다. 그런 다음 배경을 클릭하여 선택하고 속성 관리자에 [채움 색상] 선택기를 표시합니다.
7. [채움 색상] 선택기로 #00CC00 색상을 선택하여 HeaderRenderer 스킨의 배경에 적용합니다.
8. 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.
9. 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가하여 DataGrid에 데이터를 추가합니다.

```
import fl.data.DataProvider;

bldRosterGrid(aDg);
var aRoster:Array = new Array();
aRoster = [
        {Name:"Wilma Carter",Home: "Redlands, CA"},
        {Name:"Sue Pennypacker",Home: "Athens, GA"},
        {Name:"Jill Smithfield",Home: "Spokane, WA"},
        {Name:"Shirley Goth", Home: "Carson, NV"},
        {Name:"Jennifer Dunbar",Home: "Seaside, CA"}
];
aDg.dataProvider = new DataProvider(aRoster);
function bldRosterGrid(dg:DataGrid){
    dg.setSize(400, 130);
    dg.columns = ["Name", "Home"];
    dg.move(50,50);
    dg.columns[0].width = 120;
    dg.columns[1].width = 120;
};
```

10 [컨트롤] > [무비 테스트]를 선택하여 응용 프로그램을 테스트합니다.

다음 그림과 같이 머리글 행의 배경이 녹색인 DataGrid가 나타나야 합니다.

Name	Home
Wilma Carter	Redlands, CA
Sue Pennypacker	Athens, GA
Jill Smithfield	Spokane, WA
Shirley Goth	Carson, NV
Jennifer Dunbar	Seaside, CA

머리글 행 배경이 사용자 정의된 DataGrid

Label 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 Label 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. autoSize 제작 매개 변수를 설정할 수도 있습니다. 이 매개 변수를 설정해도 실시간 미리 보기의 경계 상자는 변경되지 않지만 Label의 크기는 조절됩니다. Label은 wordwrap 매개 변수에 따라 크기가 조절됩니다. 이 매개 변수가 true이면 Label은 텍스트에 맞게 세로로 크기가 조절됩니다. 이 매개 변수가 false이면 Label은 가로로 크기가 조절됩니다. 런타임에는 setSize() 메서드를 사용합니다. 자세한 내용은 **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**의 Label.setSize() 메서드 및 Label.autoSize 속성을 참조하십시오. 57페이지의 "Label 구성 요소를 사용하여 응용 프로그램 만들기"도 참조하십시오.

Label 구성 요소에 스타일 사용

스타일 속성을 설정하여 Label 인스턴스의 모양을 변경할 수 있습니다. Label 구성 요소 인스턴스의 모든 텍스트는 같은 스타일을 공유해야 합니다. Label 구성 요소에는 textFormat 스타일이 있는데, TextFormat 객체와 동일한 특성을 갖는 이 스타일을 사용하면 Label.text 내용에 일반적인 Flash TextField와 동일한 속성을 설정할 수 있습니다. 다음 예제에서는 레이블의 텍스트 색상을 빨강으로 설정합니다.

1 [구성 요소] 패널의 Label 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 a_label로 지정합니다.

2 [매개 변수] 탭을 클릭하고 텍스트 속성 값을 다음 텍스트로 변경합니다.

Color me red

3 기본 타임라인의 프레임 1을 선택하고 [액션] 패널을 연 후 다음 코드를 입력합니다.

```
/* Create a new TextFormat object, which allows you to set multiple text properties at a time. */

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
/* Apply this specific text format (red text) to the Label instance. */
a_label.setStyle("textFormat", tf);
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

Label 스타일에 대한 자세한 내용은 Flash Professional용 **ActionScript 3.0 참조 설명서**에서 Label 클래스를 참조하십시오.

스킨 및 Label

Label 구성 요소에는 스키닝할 시각적 요소가 없습니다.

List 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 List 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 List 클래스 속성을 사용합니다.

목록의 크기가 조절될 때 목록의 행이 가로로 축소되면 그 길이를 넘는 텍스트가 잘립니다. 세로일 경우에는 목록에서 필요한 만큼 행이 추가되거나 제거됩니다. 필요한 경우 스크롤 막대가 자동으로 배치됩니다.

List 구성 요소에 스타일 사용

스타일 속성을 설정하여 List 구성 요소의 모양을 변경할 수 있습니다. 스타일에 따라 구성 요소를 그릴 때 사용되는 구성 요소의 스킨 및 패딩 값이 결정됩니다.

다양한 스킨 스타일을 사용하여 스킨에 사용할 여러 클래스를 지정할 수 있습니다. 스킨 스타일 사용에 대한 자세한 내용은 91 페이지의 “스킨”을 참조하십시오.

다음 절차에서는 List 구성 요소의 contentPadding 스타일 값을 설정합니다. 내용 주위의 패딩은 List 크기에서 이 설정 값을 뺀 것이므로 List의 텍스트가 잘리지 않게 List 크기를 늘려야 할 수 있습니다.

1 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.

2 [구성 요소] 패널의 List 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aList**로 지정합니다.

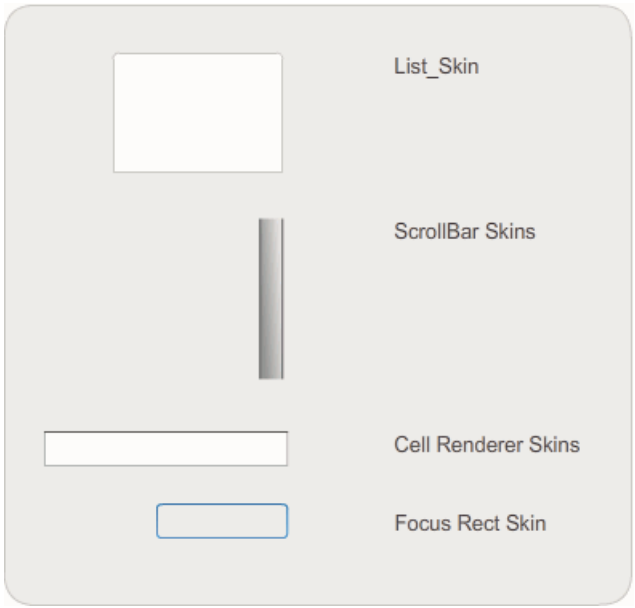
3 기본 타임라인에서 프레임 1을 선택하고 [액션] 패널을 연 후 다음 코드를 입력하여 contentPadding 스타일을 설정하고 List에 데이터를 추가합니다.

```
aList.setStyle("contentPadding", 5);
aList.setSize(145, 200);
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.rowCount = aList.length;
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

List 구성 요소에 스킨 사용

List 구성 요소는 다음과 같은 스킨을 사용하여 시각적 상태를 표현합니다.

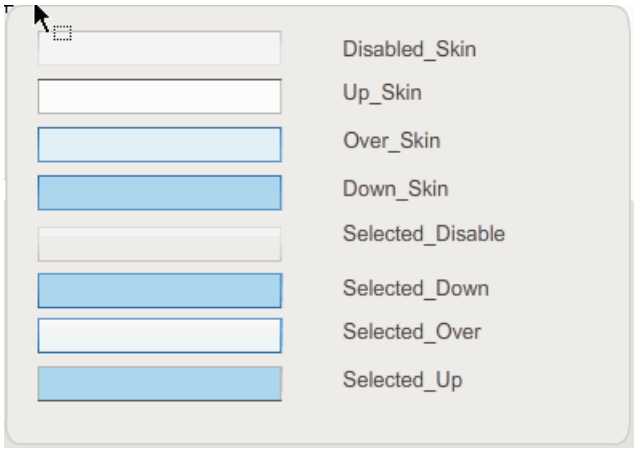


List 스킨

ScrollBar 스키닝에 대한 자세한 내용은 120페이지의 “UIScrollBar 구성 요소 사용자 정의”을 참조하십시오. Focus Rect 스킨의 스키닝에 대한 자세한 내용은 115페이지의 “TextArea 구성 요소 사용자 정의”을 참조하십시오.

참고: 한 구성 요소에서 ScrollBar 스킨을 변경하면 해당 ScrollBar를 사용하는 다른 구성 요소에서도 해당 스킨이 모두 변경됩니다.

Cell Renderer 스킨을 두 번 클릭하여 List 셀의 여러 상태에 대한 두 번째 스킨 팔레트를 엽니다.



List Cell Renderer 스킨

이러한 스킨을 편집하여 List의 셀 모양을 변경할 수 있습니다. 다음 절차에서는 Up 스킨의 색상을 변경하여 일반적인 비활성 상태의 List 모양을 변경합니다.

1. 새 Flash 파일(ActionScript 3.0) 문서를 만듭니다.
2. [구성 요소] 패널의 List 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **aList**로 지정합니다.
3. List를 두 번 클릭하여 해당 스킨 팔레트를 엽니다.
4. Cell Renderer 스킨을 두 번 클릭하여 Cell Renderer 스킨 팔레트를 엽니다.

5 Up_Skin 스킨을 두 번 클릭하여 편집할 수 있도록 엽니다.

6 스킨의 채움 영역을 클릭하여 선택합니다. [채움 색상] 선택기가 스킨의 현재 채움 색상으로 속성 관리자에 나타나야 합니다.

7 [채움 색상] 선택기로 #CC66FF 색상을 선택하여 Up_Skin 스킨의 채움에 적용합니다.

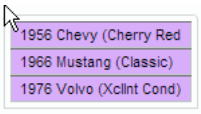
8 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

9 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가하여 List에 데이터를 추가합니다.

```
aList.setStyle("contentPadding", 5);
aList.setSize(145, 200);
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.rowCount = aList.length;
```

10 [컨트롤] > [무비 테스트]를 선택합니다.

List가 다음 그림과 같이 나타나야 합니다.



Up_Skin 색상이 사용자 정의된 List 셀

프레임 효과는 contentPadding 스타일 설정 때문입니다.

NumericStepper 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 NumericStepper 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 width, height, scaleX, scaleY 등의 해당 NumericStepper 클래스 속성과 메서드를 사용합니다.

NumericStepper 구성 요소의 크기를 조절해도 아래쪽 및 위쪽 화살표 버튼의 폭은 변경되지 않습니다. 스태퍼 크기를 기본 높이보다 크게 조절하면 기본적으로 화살표 버튼이 구성 요소의 위쪽과 아래쪽에 고정됩니다. 그렇지 않으면 9-슬라이스 크기 조절에 따라 버튼 그리기 방식이 결정됩니다. 화살표 버튼은 항상 텍스트 상자의 오른쪽에 표시됩니다.

스타일 및 NumericStepper 구성 요소

NumericStepper 구성 요소의 스타일 속성을 설정하여 모양을 변경할 수 있습니다. 스타일에 따라 구성 요소를 그릴 때 사용되는 구성 요소의 스킨, 패딩 및 텍스트 서식 값이 결정됩니다. textFormat 스타일을 사용하면 NumericStepper 값의 크기와 모양을 변경할 수 있습니다. 다양한 스킨 스타일을 사용하여 스킨에 사용할 여러 클래스를 지정할 수 있습니다. 스킨 스타일 사용에 대한 자세한 내용은 91페이지의 “스킨”을 참조하십시오.

이 절차에서는 textFormat 스타일을 사용하여 NumericStepper에서 표시하는 값의 모양을 변경합니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

2 [구성 요소] 패널의 NumericStepper 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **myNs**로 지정합니다.

3 기본 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

```
var tf:TextFormat = new TextFormat();
myNs.setSize(100, 50);
tf.color = 0x0000CC;
tf.size = 24;
tf.font = "Arial";
tf.align = "center";
myNs.setStyle("textFormat", tf);
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

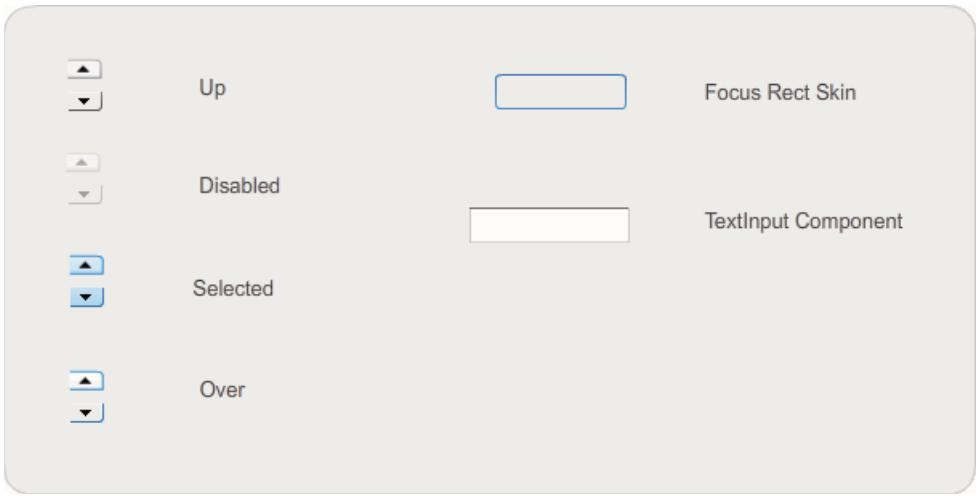
스킨 및 NumericStepper 구성 요소

NumericStepper 구성 요소에는 버튼의 업, 다운, 비활성화 및 선택 상태를 나타내는 스킨이 있습니다.

스태퍼가 활성화된 경우에는 포인터가 그 위를 이동할 때 아래쪽 및 위쪽 버튼이 오버 상태로 표시됩니다. 버튼을 누르면 다운 상태로 표시됩니다. 마우스를 놓으면 버튼은 오버 상태로 되돌아갑니다. 마우스를 누른 상태에서 포인터를 버튼 밖으로 이동하면 버튼은 원래 상태로 되돌아갑니다.

스태퍼가 비활성화된 경우에는 사용자 상호 작용과 상관없이 비활성 상태로 표시됩니다.

NumericStepper 구성 요소에는 다음과 같은 스킨이 있습니다.



NumericStepper 스킨

1 새 FLA 파일을 만듭니다.

2 NumericStepper 구성 요소를 스테이지로 드래그합니다.

3 확대/축소 컨트롤을 400%로 설정하여 이미지를 편집하기 쉽게 확대합니다.

4 스킨 패널에서 TextInput 스킨의 배경을 두 번 클릭하여 그룹 수준까지 드릴다운하고 속성 관리자의 [채움 색상] 선택기에 배경색을 표시합니다.

5 속성 관리자의 [채움 색상] 선택기로 #9999FF 색상을 선택하여 TextInput 스킨의 배경에 적용합니다.

6 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

7 NumericStepper를 다시 두 번 클릭하여 스킨 패널을 다시 엽니다.

8 [Up] 그룹에서 위쪽 화살표 버튼의 배경을 두 번 클릭하여 선택하고 속성 관리자의 [채움 색상] 선택기에 해당 색상을 표시합니다.

9 #9966FF 색상을 선택하여 위쪽 화살표 버튼의 배경에 적용합니다.

10 [Up] 그룹의 아래쪽 화살표에 대해서도 8~9단계를 반복합니다.

11 [컨트롤] > [무비 테스트]를 선택합니다.

다음 그림과 같이 NumericStepper 인스턴스가 나타나야 합니다.

ProgressBar 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 ProgressBar 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 ProgressBar 클래스 속성을 사용합니다.

ProgressBar에는 트랙 스킨, 막대 스킨, 불확정 스킨의 세 가지 스킨이 있으며, 9-슬라이스 크기 조절을 사용하여 에셋의 크기를 조절합니다.

스타일 및 ProgressBar 구성 요소

스타일 속성을 설정하여 ProgressBar 인스턴스의 모양을 변경할 수 있습니다. ProgressBar의 스타일에 따라 구성 요소를 그릴 때 사용되는 구성 요소의 스킨 및 패딩 값이 결정됩니다. 다음 예제에서는 ProgressBar 인스턴스의 크기를 확대하고 barPadding 스타일을 설정합니다.

1 새 FLA 파일을 만듭니다.

2 [구성 요소] 패널의 ProgressBar 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **myPb**로 지정합니다.

3 기본 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 입력합니다.

```
myPb.width = 300;
myPb.height = 30;

myPb.setStyle("barPadding", 3);
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

스킨 스타일 설정에 대한 자세한 내용은 91페이지의 “스킨”을 참조하십시오.

스킨 및 ProgressBar 구성 요소

ProgressBar 구성 요소는 다음 그림과 같이 스킨을 사용하여 진행률 막대 트랙, 완료된 막대 및 불확정 막대를 나타냅니다.

ProgressBar 스킨

막대는 barPadding으로 위치를 결정하여 트랙 스킨 위에 배치됩니다. 에셋의 크기는 9-슬라이스 크기 조절을 사용하여 조절됩니다.

불확정 막대는 ProgressBar 인스턴스의 indeterminate 속성이 true로 설정된 경우 사용됩니다. 이 스킨은 ProgressBar 크기에 맞게 가로/세로로 크기가 조절됩니다.

이러한 스킨을 편집하여 ProgressBar 모양을 변경할 수 있습니다. 예를 들어, 다음 예제에서는 불확정 막대의 색상을 변경합니다.

1 새 FLA 파일을 만듭니다.

2 ProgressBar 구성 요소를 스테이지로 드래그하고 두 번 클릭하여 스킨 아이콘 패널을 엽니다.

3 불확정 막대 스킨을 두 번 클릭합니다.

4 확대/축소 컨트롤을 400%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.

5 대각선 막대 중 하나를 두 번 클릭하고 Shift 키를 누른 채로 나머지 대각선 막대를 각각 클릭합니다. 현재 색상이 속성 관리자의 [채움 색상] 선택기에 나타납니다.

6 속성 관리자의 [채움 색상] 선택기를 클릭하여 열고 #00CC00 색상을 선택하여 앞에서 선택한 대각선 막대에 적용합니다.

7 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

8 [컨트롤] > [무비 테스트]를 선택합니다.

ProgressBar가 다음 그림과 같이 나타나야 합니다.

RadioButton 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 RadioButton 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용합니다.

RadioButton 구성 요소의 경계 상자는 표시되지 않으며 구성 요소에 대한 히트 영역을 지정합니다. 구성 요소의 크기를 늘리면 히트 영역의 크기도 늘어납니다.

구성 요소 레이블에 비해 구성 요소의 경계 상자가 작으면 경계 상자에 맞게 레이블이 잘립니다.

RadioButton 구성 요소에 스타일 사용

스타일 속성을 설정하여 RadioButton의 모양을 변경할 수 있습니다. RadioButton의 스타일 속성에 따라 구성 요소를 그릴 때 사용되는 스킨, 아이콘, 텍스트 서식 및 패딩 값이 결정되고, 레이아웃으로 사용되는 스킨 및 패딩 값이 결정됩니다.

다음 예제에서는 CheckBox 구성 요소에서 textFormat 스타일을 가져와 RadioButton에 적용하여 레이블을 동일한 스타일로 만듭니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

2 CheckBox 구성 요소를 스테이지로 드래그하고 속성 관리자에서 인스턴스 이름을 **myCh**로 지정합니다.

3 RadioButton을 스테이지로 드래그하고 속성 관리자에서 인스턴스 이름을 **myRb**로 지정합니다.

4 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

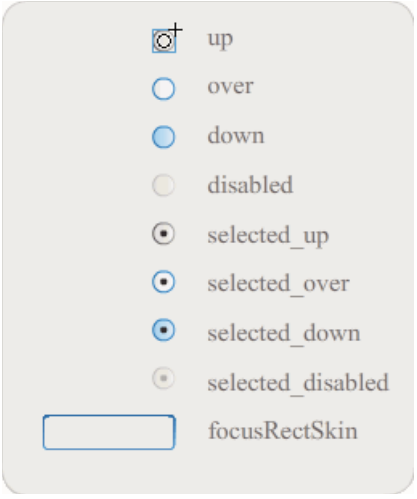
```
var tf:TextFormat = new TextFormat();
tf.color = 0x00FF00;
tf.font = "Georgia";
tf.size = 18;
myCh.setStyle("textFormat", tf);
myRb.setStyle("textFormat", myCh.getStyle("textFormat"));
```

이 코드에서는 CheckBox의 textFormat 스타일을 설정하고 CheckBox에서 getStyle() 메서드를 호출하여 RadioButton에 해당 스타일을 적용합니다.

5 [컨트롤] > [무비 테스트]를 선택합니다.

스킨 및 RadioButton 구성 요소

RadioButton에는 다음과 같은 스킨이 있으며, 이러한 스킨을 편집하여 모양을 변경할 수 있습니다.



RadioButton 스킨

RadioButton이 활성화되었으나 선택되지 않은 경우 사용자가 RadioButton 위로 포인터를 이동하면 over 스킨이 표시됩니다. 사용자가 RadioButton을 클릭하면 RadioButton이 입력 포커스를 받고 selected_down 스킨이 표시됩니다. 사용자가 마우스를 놓으면 RadioButton에 selected_up 스킨이 표시됩니다. 사용자가 마우스 버튼을 누른 채 RadioButton의 히트 영역 밖으로 포인터를 이동하면 RadioButton에 up 스킨이 다시 표시됩니다.

RadioButton이 비활성화된 경우에는 사용자 상호 작용과 상관없이 비활성 상태로 표시됩니다.

다음 예제에서는 선택 상태를 나타내는 selected_up 스킨을 바꿉니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

2 RadioButton 구성 요소를 스테이지로 드래그하고 두 번 클릭하여 해당 스킨 팔레트를 엽니다.

3 확대/축소 컨트롤을 800%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.

4 selected_up 스킨을 두 번 클릭하여 선택하고 Delete 키를 눌러 삭제합니다.

5 [도구] 패널에서 [사각형 도구]를 선택합니다.

6 속성 관리자에서 선 색상을 빨강(#FF0000)으로 설정하고 채움 색상을 검정(#000000)으로 설정합니다.

7 포인터를 클릭하고 드래그하여 사각형을 그립니다. 이때 심볼의 등록 포인트(원점이라고도 함)를 나타내는 십자형에서 시작합니다.

8 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

9 [컨트롤] > [무비 테스트]를 선택합니다.

10 RadioButton을 클릭하여 선택합니다.

선택 상태의 RadioButton이 다음 그림과 유사하게 나타나야 합니다.

ScrollPane 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 ScrollPane 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 ScrollPane 클래스 속성과 메서드를 사용합니다.

ScrollPane 구성 요소에는 다음과 같은 그래픽 특성이 있습니다.

- 내용의 등록 포인트(원점이라고도 함)가 창의 왼쪽 위 모서리에 있습니다.
- 가로 스크롤 막대를 해제하면 세로 스크롤 막대가 스크롤 창의 오른쪽에 위에서 아래 방향으로 표시됩니다. 세로 스크롤 막대를 해제하면 가로 스크롤 막대가 스크롤 창의 아래쪽에 왼쪽에서 오른쪽 방향으로 표시됩니다. 두 가지 스크롤 막대 모두를 숨길 수 있습니다.
- 스크롤 창이 너무 작으면 내용이 제대로 표시되지 않을 수 있습니다.
- 스크롤 창 크기를 조절하면 스크롤 트랙과 스크롤 상자(썸)가 확장되거나 축소되고 해당 히트 영역의 크기가 조절됩니다. 버튼 크기는 그대로 유지됩니다.

ScrollPane 구성 요소에 스타일 사용

ScrollPane 구성 요소의 스타일 속성에 따라 구성 요소를 그릴 때 레이아웃으로 사용되는 스킨 및 패딩 값이 결정됩니다. 다양한 스킨 스타일을 사용하여 구성 요소마다 다른 클래스를 지정하여 사용할 수 있습니다. 스킨 스타일 사용에 대한 자세한 내용은 91페이지의 “스킨”을 참조하십시오.

1. 새 Flash 문서(ActionScript 3.0)를 만듭니다.
2. ScrollPane 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **mySp**로 지정합니다.
3. 속성 관리자에서 [매개 변수] 탭을 클릭하고 source 매개 변수에 **http://www.helpexamples.com/flash/images/image1.jpg** 값을 입력합니다.
4. 기본 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

   ```
   mySp.setStyle("contentPadding", 5);
   ```

   패딩은 스크롤 막대 바깥쪽에서 구성 요소의 테두리와 내용 사이에 적용됩니다.
5. [컨트롤] > [무비 테스트]를 선택합니다.

스킨 및 ScrollPane

ScrollPane 구성 요소는 스크롤 에셋으로 테두리와 스크롤 막대를 사용합니다. 스크롤 막대 스키닝에 대한 자세한 내용은 120페이지의 “UIScrollBar 구성 요소에 스킨 사용”을 참조하십시오.

Slider 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 Slider 구성 요소를 가로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 width, scaleX 등의 해당 Slider 클래스 속성을 사용합니다.

슬라이더는 길이만 늘릴 수 있고 높이는 늘릴 수 없습니다. height 속성과 setSize() 메서드의 height 매개 변수는 무시됩니다. 물론, 세로 방향의 슬라이더를 만들어 세로 방향으로 길게 만들 수는 있습니다.

스타일 및 Slider 구성 요소

Slider 구성 요소의 스타일은 스킨 클래스와, 구성 요소 경계 상자와 구성 요소 외부 경계 간의 패딩에 사용할 픽셀 수를 지정하는 FocusRectPadding 값만 결정합니다. 스킨 스타일 사용에 대한 자세한 내용은 91페이지의 “스킨”을 참조하십시오.

스킨 및 Slider 구성 요소

Slider 구성 요소에는 다음과 같은 스킨이 있으며, 이러한 스킨을 편집하여 모양을 변경할 수 있습니다.

Slider 스킨

다음 예제에서는 up 트랙을 편집하여 색상을 파랑으로 변경합니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

2 [구성 요소] 패널의 Slider 구성 요소를 스테이지로 드래그합니다.

3 Slider 구성 요소를 두 번 클릭하여 해당 패널을 엽니다.

4 up 트랙의 등록 표시를 두 번 클릭하여 심볼 편집 모드로 엽니다.

5 확대/축소 컨트롤을 800%로 설정하여 아이콘을 편집하기 쉽게 확대합니다. Slider의 트랙은 세 가지 막대로 구성되어 있습니다.

6 위쪽 막대를 클릭하여 선택합니다. 그러면 막대 색상이 속성 관리자의 [채움 색상] 선택기에 표시됩니다.

7 속성 관리자의 [채움 색상] 선택기로 #000066 색상을 선택하여 Slider 트랙의 위쪽 막대에 적용합니다.

8 Slider 트랙의 가운데 막대를 클릭하여 선택합니다. 그러면 막대 색상이 속성 관리자의 [채움 색상] 선택기에 표시됩니다.

9 속성 관리자의 [채움 색상] 선택기로 #0066FF 색상을 선택하여 Slider 트랙의 가운데 막대에 적용합니다.

10 Slider 트랙의 아래쪽 막대를 클릭하여 선택합니다. 그러면 막대 색상이 속성 관리자의 [채움 색상] 선택기에 표시됩니다.

11 속성 관리자의 [채움 색상] 선택기로 #00CCFF 색상을 선택하여 Slider 트랙의 아래쪽 막대에 적용합니다.

12 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

13 [컨트롤] > [무비 테스트]를 선택합니다.

Slider가 다음 그림과 같이 나타나야 합니다.

TextArea 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 TextArea 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 TextArea 클래스 속성을 사용합니다.

TextArea 구성 요소의 크기를 조절하면 테두리 크기가 새 경계 상자에 맞게 조절됩니다. 필요하면 스크롤 막대가 아래쪽과 오른쪽 가장자리에 배치되고 나머지 영역 안에 텍스트 영역의 크기가 조절됩니다. TextArea 구성 요소에 있는 모든 요소의 크기는 고정되어 있지 않습니다. TextArea 구성 요소 폭이 텍스트 크기에 비해 너무 좁으면 텍스트가 잘립니다.

스타일 및 TextArea 구성 요소

TextArea 구성 요소의 스타일에 따라 구성 요소를 그릴 때 사용되는 스킨, 패딩 및 텍스트 서식 값이 결정됩니다. texFormat 및 disabledTextFormat 스타일에 따라 TextArea가 표시하는 텍스트의 스타일이 달라집니다. 스킨 스타일 속성에 대한 자세한 내용은 115페이지의 “TextArea 구성 요소에 스킨 사용”을 참조하십시오.

다음 예제에서는 disabledTextFormat 스타일을 설정하여 TextArea가 비활성 상태일 때의 텍스트 모양을 변경하지만 동일한 절차를 사용하여 활성 상태의 TextArea에 대한 textFormat 스타일도 설정할 수 있습니다.

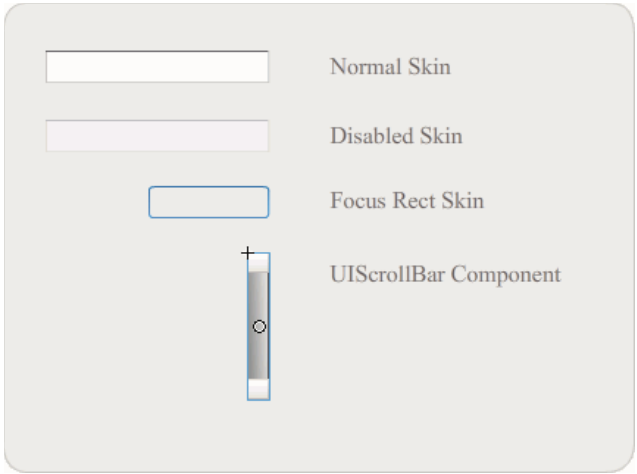
1. 새 Flash 파일을 만듭니다.
2. TextArea 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **myTa**로 지정합니다.
3. 기본 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

```
var tf:TextFormat = new TextFormat();
tf.color = 0xCC99FF;
tf.font = "Arial Narrow";
tf.size = 24;
myTa.setStyle("disabledTextFormat", tf);
myTa.text = "Hello World";
myTa.setSize(120, 50);
myTa.move(200, 50);
myTa.enabled = false;
```

4. [컨트롤] > [무비 테스트]를 선택합니다.

TextArea 구성 요소에 스킨 사용

TextArea 구성 요소에는 다음과 같은 스킨이 있으며, 이러한 스킨을 편집하여 모양을 변경할 수 있습니다.



TextArea 스킨

참고: 한 구성 요소에서 ScrollBar 스킨을 변경하면 해당 ScrollBar를 사용하는 다른 구성 요소에서도 해당 스킨이 모두 변경됩니다.

다음 절차에서는 TextArea에 포커스가 있으며 Normal 스킨 상태일 때 나타나는 Focus Rect 스킨의 테두리 색상을 변경합니다.

1 새 Flash 파일을 만듭니다.

2 TextArea 구성 요소를 스테이지로 드래그하고 두 번 클릭하여 스킨 아이콘 패널을 엽니다.

3 Focus Rect 스킨을 두 번 클릭합니다.

4 Focus Rect 스킨의 테두리를 클릭하여 선택합니다. 그러면 막대 색상이 속성 관리자의 [채움 색상] 선택기에 표시됩니다.

5 속성 관리자의 [채움 색상] 선택기를 클릭하여 열고 #CC0000 색상을 선택하여 테두리에 적용합니다.

6 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

7 TextArea 구성 요소를 두 번 클릭하여 스킨 아이콘 패널을 엽니다.

8 Normal 스킨을 두 번 클릭합니다.

9 Normal 스킨 테두리의 각 가장자리를 한 번에 하나씩 선택하여 색상을 #990099로 설정합니다.

10 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

11 [컨트롤] > [무비 테스트]를 선택합니다.

TextArea를 선택하고 텍스트 입력을 시작하면 테두리가 다음 그림과 같이 나타나야 합니다.

바깥쪽 테두리는 Focus Rect 스킨이며 안쪽 테두리는 Normal 스킨의 테두리입니다.

UIScrollBar 스킨 편집에 대한 자세한 내용은 120페이지의 “UIScrollBar 구성 요소 사용자 정의”을 참조하십시오.

TextInput 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 TextInput 인스턴스의 크기를 변경할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 height, width, scaleX, scaleY 등의 해당 TextInput 클래스 속성을 사용합니다.

TextInput 구성 요소의 크기를 조절하면 테두리 크기가 새 경계 상자에 맞게 조절됩니다. TextInput 구성 요소는 스크롤 막대를 사용하지 않지만 사용자가 텍스트와 상호 작용하면 삽입점이 자동으로 스크롤됩니다. 나머지 영역 안에 텍스트 필드의 크기가 조절됩니다. TextInput 구성 요소에 있는 모든 요소의 크기는 고정되어 있지 않습니다. TextInput 구성 요소가 너무 작으면 텍스트가 잘립니다.

스타일 및 TextInput 구성 요소

TextInput 구성 요소의 스타일에 따라 구성 요소를 그릴 때 사용되는 스킨, 패딩 및 텍스트 서식 값이 결정됩니다. texFormat 및 disabledTextFormat 스타일에 따라 구성 요소에 표시되는 텍스트의 스타일이 달라집니다. 스킨 스타일 속성에 대한 자세한 내용은 117페이지의 “스킨 및 TextInput 구성 요소”을 참조하십시오.

다음 예제에서는 textFormat 스타일을 설정하여 TextInput 구성 요소에 표시되는 텍스트의 글꼴, 크기 및 색상을 설정합니다. 구성 요소가 비활성 상태일 때 적용되는 disabledTextFormat 스타일도 동일한 절차로 설정할 수 있습니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

2 TextInput 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **myTi**로 지정합니다.

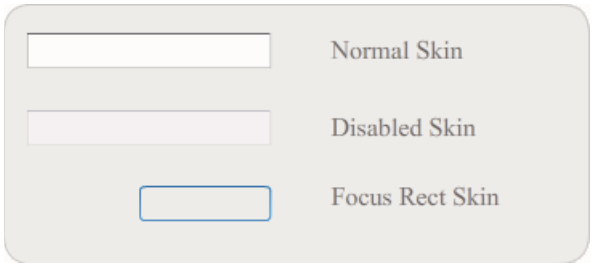
3 기본 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

```
var tf:TextFormat = new TextFormat();
tf.color = 0x0000FF;
tf.font = "Verdana";
tf.size = 30;
tf.align = "center";
tf.italic = true;
myTi.setStyle("textFormat", tf);
myTi.text = "Enter your text here";
myTi.setSize(350, 50);
myTi.move(100, 50);
```

4 [컨트롤] > [무비 테스트]를 선택합니다.

스킨 및 TextInput 구성 요소

TextInput 구성 요소에는 다음과 같은 스킨이 있으며, 이러한 스킨을 편집하여 모양을 변경할 수 있습니다.



TextInput 캡션

다음 절차에서는 TextInput 구성 요소의 테두리와 배경색을 변경합니다.

1 새 Flash 파일을 만듭니다.

2 TextInput 구성 요소를 스테이지로 드래그하고 두 번 클릭하여 스킨 패널을 엽니다.

3 Normal 스킨을 두 번 클릭합니다.

4 확대/축소 컨트롤을 800%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.

5 Normal 스킨 테두리의 각 가장자리를 한 번에 하나씩 선택하여 해당 색상을 #993399로 설정하고 적용합니다.

6 배경을 두 번 클릭하여 속성 관리자의 [채움 색상] 선택기에 배경색을 표시합니다. #99CCCC 색상을 선택하여 배경색에 적용합니다.

7 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

8 [컨트롤] > [무비 테스트]를 선택합니다.

TextInput 구성 요소가 다음 그림과 같이 나타나야 합니다.

TileList 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 TileList 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 width, height, columnCount, rowCount, scaleX, scaleY 등의 해당 속성을 사용합니다. TileList에 포함된 ScrollBar는 목록 상자와 함께 크기가 조절됩니다.

스타일 및 TileList 구성 요소

TileList의 스타일에 따라 구성 요소를 그릴 때 사용되는 스킨, 패딩 및 텍스트 서식 값이 결정됩니다. texFormat 및 disabledTextFormat 스타일에 따라 구성 요소에 표시되는 텍스트의 스타일이 달라집니다. 스킨 스타일에 대한 자세한 내용은 118 페이지의 “TileList 구성 요소에 스킨 사용”을 참조하십시오.

다음 예제에서는 textFormat 스타일을 사용하여 TileList 인스턴스에 표시되는 레이블의 글꼴, 크기, 색상 및 텍스트 특성을 설정하는 setRendererStyle() 메서드를 호출합니다. 동일한 절차를 수행하여 enabled 속성이 false로 설정되어 있는 경우 적용되는 disabledTextFormat 스타일도 설정할 수 있습니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

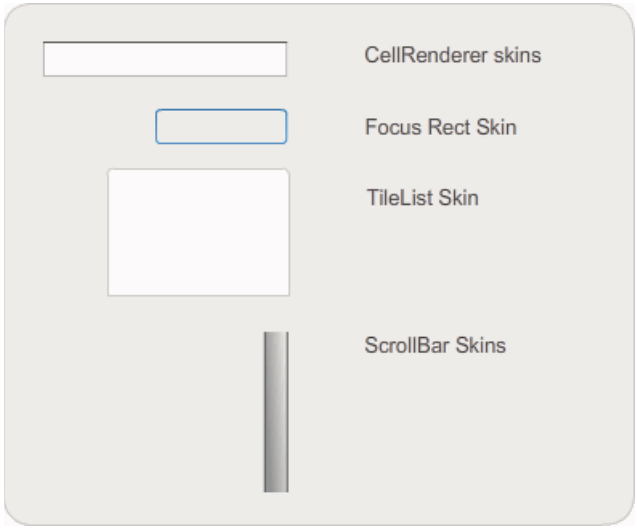
2 TileList 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **myTl**로 지정합니다.

3 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

```
myTl.setSize(100, 100);
myTl.addItem({label:"#1"});
myTl.addItem({label:"#2"});
myTl.addItem({label:"#3"});
myTl.addItem({label:"#4"});
var tf:TextFormat = new TextFormat();
tf.font = "Arial";
tf.color = 0x00FF00;
tf.size = 16;
tf.italic = true;
tf.bold = true;
tf.underline = true;
tf.align = "center";
myTl.setRendererStyle("textFormat", tf);
```

TileList 구성 요소에 스킨 사용

TileList 구성 요소에는 TileList 스킨, CellRenderer 스킨 및 ScrollBar 스킨이 있습니다. 이러한 스킨을 편집하여 TileList의 모양을 변경할 수 있습니다.



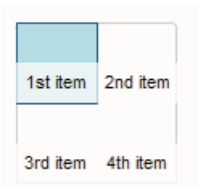
TileList 스킨

참고: 한 구성 요소에서 ScrollBar 스킨을 변경하면 해당 ScrollBar를 사용하는 다른 구성 요소에서도 해당 스킨이 모두 변경됩니다.

다음 절차에서는 TileList의 CellRenderer Selected_Up 스킨의 색상을 변경합니다.

1. Flash 문서(ActionScript 3.0)를 만듭니다.
2. TileList 구성 요소를 스테이지로 드래그하고 두 번 클릭하여 스킨 패널을 엽니다.
3. CellRenderer 스킨을 두 번 클릭하고 Selected_Up 스킨을 두 번 클릭한 후 사각형 배경을 클릭합니다.
4. 속성 관리자의 [채움 색상] 선택기로 #99FFFF 색상을 선택하여 Selected_Up 스킨에 적용합니다.
5. 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.
6. 속성 관리자의 [매개 변수] 탭에서 dataProvider 행의 두 번째 열을 두 번 클릭하여 [값] 대화 상자를 엽니다. 1st item, 2nd item, 3rd item, 4th item이라는 레이블을 사용하여 항목을 추가합니다.
7. [컨트롤] > [무비 테스트]를 선택합니다.
8. TileList에서 셀 하나를 클릭하여 선택한 다음 선택한 셀 외부로 마우스를 이동합니다.

   선택한 셀이 다음 그림과 같이 나타나야 합니다.



Selected_Up 스킨 색상이 변경된 TileList 구성 요소

UILoader 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 UILoader 구성 요소를 가로/세로로 변형할 수 있습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 width, height, scaleX, scaleY 등의 해당 속성을 사용합니다.

UILoader 구성 요소의 크기 조절 비헤이비어는 scaleContent 속성을 통해 제어됩니다. scaleContent가 true인 경우 내용의 크기는 로더에 맞게 조절되고 setSize()가 호출될 때 다시 조절됩니다. scaleContent가 false인 경우에는 구성 요소의 크기가 내용의 크기에 맞게 고정되고 setSize()와 크기 조절 속성은 영향을 미치지 않습니다.

UILoader 구성 요소에는 스타일이나 스킨을 적용할 수 있는 사용자 인터페이스 요소가 없습니다.

UIScrollBar 구성 요소 사용자 정의

제작하는 동안 또는 런타임에 UIScrollBar 구성 요소를 가로/세로로 변형할 수 있습니다. 그렇지만, 세로 UIScrollBar는 폭을 수정할 수 없으며, 세로 UIScrollBar는 높이를 수정할 수 없습니다. 제작하는 동안에는 스테이지에서 구성 요소를 선택한 다음 [자유 변형 도구]나 [수정] > [변형] 명령을 사용합니다. 런타임에는 setSize() 메서드를 사용하거나 width, height, scaleX, scaleY 등의 해당 UIScrollBar 클래스 속성을 사용합니다.

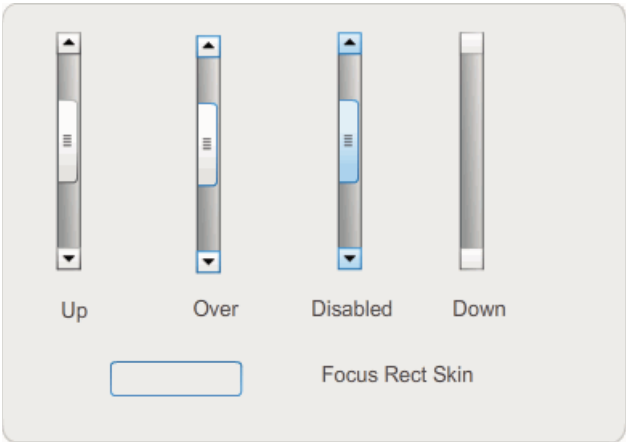
참고: setSize() 메서드를 사용하는 경우 가로 스크롤 막대의 폭이나 세로 스크롤 막대의 높이만 변경할 수 있습니다. 제작 시 가로 스크롤 막대의 높이나 세로 스크롤 막대의 폭을 설정할 수 있지만 무비를 제작하면 값이 재설정됩니다. 길이에 해당되는 스크롤 막대의 치수만 변경할 수 있습니다.

UIScrollBar 구성 요소에 스타일 사용

UIScrollBar 구성 요소의 스타일은 스킨 클래스와, 구성 요소 경계 상자와 구성 요소 외부 경계 간의 패딩에 사용할 픽셀 수를 지정하는 FocusRectPadding 값만 결정합니다. 스킨 스타일 사용에 대한 자세한 내용은 91페이지의 "스킨"을 참조하십시오.

UIScrollBar 구성 요소에 스킨 사용

UIScrollBar 구성 요소는 다음과 같은 스킨을 사용합니다.



UIScrollBar 스킨

가로 스크롤 막대와 세로 스크롤 막대 모두 동일한 스킨을 사용하며 가로 스크롤 막대를 표시할 때는 UIScrollBar 구성 요소에서 스킨을 적절히 회전합니다.

참고: 한 구성 요소에서 ScrollBar 스킨을 변경하면 해당 ScrollBar를 사용하는 다른 구성 요소에서도 해당 스킨이 모두 변경됩니다.

다음 예제에서는 UIScrollBar의 썸과 화살표 버튼의 색상을 변경하는 방법을 보여 줍니다.

1 새 Flash 문서(ActionScript 3.0)를 만듭니다.

2 UIScrollBar 구성 요소를 스테이지로 드래그하고 인스턴스 이름을 **mySb**로 지정합니다. [매개 변수] 탭에서 방향을 가로로 설정합니다.

3 스크롤 막대를 두 번 클릭하여 스킨 패널을 엽니다.

4 Up 스킨을 클릭하여 선택합니다.

5 확대/축소 컨트롤을 400%로 설정하여 아이콘을 편집하기 쉽게 확대합니다.

6 오른쪽 화살표(또는 세로 스크롤 막대의 위쪽 화살표) 배경을 두 번 클릭하여 선택하고 속성 관리자의 [채움 색상] 선택기에 해당 색상을 표시합니다.

7 #CC0033 색상을 선택하여 버튼 배경색에 적용합니다.

8 스테이지 위의 편집 막대 왼쪽에 있는 [뒤로] 버튼을 클릭하여 문서 편집 모드로 돌아갑니다.

9 썸 및 왼쪽 화살표(또는 세로 스크롤 막대의 아래쪽 화살표) 요소에 대해서도 6~8단계를 반복합니다.

10 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가하여 스크롤 막대를 TextField에 첨부합니다.

```
var tf:TextField = new TextField();
addChild(tf);
tf.x = 150;
tf.y = 100;
mySb.width = tf.width = 200;
tf.height = 22;
tf.text = "All work and no play makes Jack a dull boy. All work and no play makes Jack a dull boy. All
. . .";
mySb.y = tf.y + tf.height;
mySb.x = tf.x + tf.width;x
mySb.scrollTarget = tf;
```

11 [컨트롤] > [무비 테스트]를 선택합니다.

UIScrollBar 구성 요소가 다음 그림과 같이 나타나야 합니다.

썸과 왼쪽 및 오른쪽 화살표가 빨강으로 표시된 가로 스크롤 막대

6장: FLVPlayback 구성 요소 사용

FLVPlayback 구성 요소를 사용하면 Adobe Flash CS5 Professional 응용 프로그램에 비디오 플레이어를 쉽게 추가할 수 있으므로 점진적으로 다운로드되는 비디오 파일을 HTTP를 통해 재생하거나 Adobe의 Macromedia Flash Media Server 또는 FVSS(Flash Video Streaming Service)에서 스트리밍 FLV 파일을 재생할 수 있습니다.

Adobe Flash Player 9 업데이트 3(버전 9.0.115.0 이상) 릴리스에서는 Flash Player에서 비디오 내용을 재생하기 위한 대폭 향상된 기능이 통합되었습니다. 이 업데이트에는 최종 사용자의 시스템 비디오 하드웨어를 사용하여 비디오 재생 성능을 향상시킬 수 있도록 FLVPlayback 구성 요소에 대한 변경 사항이 포함되어 있습니다. 또한 FLVPlayback 구성 요소의 변경 사항을 통해 비디오 파일이 전체 화면 모드에서 더욱 선명하게 표시됩니다.

Flash Player 9 업데이트 3에서는 업계 표준 H.264 인코딩을 사용하는 고화질 MPEG-4 비디오 형식에 대한 지원을 추가하여 FLVPlayback 구성 요소의 기능이 향상되었습니다. 이러한 형식에는 MP4, M4A, MOV, MP4V, 3GP 및 3G2가 포함됩니다.

참고: Apple® iTunes®에서 다운로드하거나 FairPlay®로 디지털 암호화되어 보호된 MP4 파일은 지원되지 않습니다.

사용이 간편한 FLVPlayback 구성 요소는 다음과 같은 특성과 장점을 갖고 있습니다.

- 스테이지로 드래그하여 신속하고 성공적으로 구현할 수 있음
- 전체 화면 크기 지원
- 재생 컨트롤의 모양을 사용자 정의할 수 있는 미리 제작된 **skins** 컬렉션 제공
- 미리 제작된 스킨의 색상과 알파 값을 선택할 수 있음
- 고급 사용자를 위한 스킨 제작 기능 제공
- 제작 중 실시간 미리 보기 기능 제공
- 크기 조절 시 비디오 파일이 중앙에 오도록 레이아웃 속성 제공
- 점진적으로 다운로드되는 비디오 파일을 일정 비율 이상 다운로드한 경우 재생 시작 허용
- 비디오를 텍스트, 그래픽 및 애니메이션과 동기화할 수 있는 큐 포인트 제공
- SWF 파일 크기를 적정한 규모로 유지

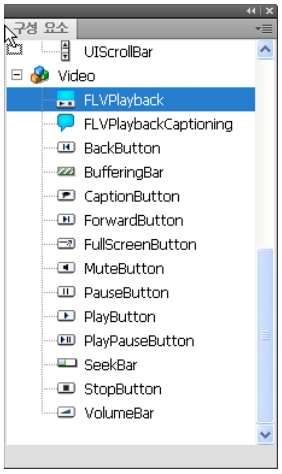
FLVPlayback 구성 요소 사용

FLVPlayback 구성 요소를 사용하는 과정은 이 구성 요소를 스테이지에 배치하고 재생할 비디오 파일을 지정하는 두 가지 작업으로 구성됩니다. 뿐만 아니라 구성 요소의 비헤이비어를 제어하고 비디오 파일을 지정하는 다양한 매개 변수를 설정할 수도 있습니다.

또한 FLVPlayback 구성 요소에는 ActionScript API(Application Programming Interface)가 포함되어 있으며, API에는 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서에 자세히 설명되어 있는 클래스(CuePointType, FLVPlayback, FLVPlaybackCaptioning, NCManager, NCManagerNative, VideoAlign, VideoError, VideoPlayer, VideoState)와 몇 가지 이벤트 클래스(AutoLayoutEvent, LayoutEvent, MetadataEvent, SkinErrorEvent, SoundEvent, VideoEvent, VideoProgressEvent)가 포함되어 있습니다.

FLVPlayback 구성 요소에는 FLV PLayback Custom UI 구성 요소가 포함되어 있습니다. FLVPlayback 구성 요소는 비디오 파일을 표시하는 디스플레이 영역(또는 비디오 플레이어)과 비디오 파일을 작동하는 컨트롤로 구성되어 있습니다. FLV Playback Custom UI 구성 요소는 비디오 파일을 재생, 중단, 일시 정지 및 기타 방식으로 제어하는 데 사용할 수 있는 컨트롤 버튼과 메커니즘을 제공합니다. 이러한 컨트롤에는 BackButton, BufferingBar, CaptionButton(FLVPlaybackCaptioning용),

ForwardButton, FullScreenButton, MuteButton, PauseButton, PlayButton, PlayPauseButton, SeekBar, StopButton, VolumeBar 등이 있습니다. FLVPlayback 구성 요소 및 FLV Playback Custom UI 컨트롤은 [구성 요소] 패널에서 다음 그림과 같이 나타납니다.



구성 요소 패널에 표시된 FLVPlayback 구성 요소

FLVPlayback 구성 요소에 재생 컨트롤을 추가하는 프로세스를 **스키닝**이라고 합니다. FLVPlayback 구성 요소에는 재생, 중단, 뒤로 감기, 앞으로 감기, 검색 막대, 음소거, 볼륨 조절, 전체 화면 및 캡션 컨트롤을 제공하는 기본 스킨인 SkinOverAll.swf가 있습니다. 이 스킨을 변경하려면 다음 중 하나를 수행합니다.

- 미리 제작된 스킨 컬렉션에서 선택
- 사용자 정의 스킨을 만들어 미리 제작된 스킨 컬렉션에 추가
- FLV Playback Custom UI 구성 요소에서 개별 컨트롤을 선택하여 사용자 정의

  미리 제작된 스킨을 선택하면 제작하는 동안이나 런타임에 스킨 색상과 알파 값을 별도로 선택할 수 있습니다. 자세한 내용은 138페이지의 “미리 제작된 스킨 선택”을 참조하십시오.

  다른 스킨을 선택하면 선택한 스킨은 새로운 기본 스킨이 됩니다.

  FLVPlayback 구성 요소의 스킨을 선택하거나 만드는 방법에 대한 자세한 내용은 138페이지의 “FLVPlayback 구성 요소 사용자 정의”를 참조하십시오.

FLVPlayback 구성 요소를 사용하여 응용 프로그램 만들기

다음과 같은 방법으로 FLVPlayback 구성 요소를 응용 프로그램에 추가할 수 있습니다.

- [구성 요소] 패널의 FLVPlayback 구성 요소를 스테이지로 드래그하고 source 매개 변수에 대한 값을 지정합니다.
- [비디오 가져오기] 마법사를 사용하여 스테이지에 구성 요소를 만들고 스킨을 선택하여 구성 요소를 사용자 정의합니다.
- FLVPlayback() 생성자를 사용하여 스테이지에 FLVPlayback 인스턴스를 동적으로 만듭니다(구성 요소가 라이브러리에 있는 경우).

  참고: ActionScript를 사용하여 FLVPlayback 인스턴스를 만드는 경우 ActionScript에서 skin 속성을 설정하여 인스턴스에 스킨을 지정해야 합니다. 이 방법으로 스킨을 적용하면 스킨이 SWF 파일로 자동 제작되지 않으므로 응용 프로그램 SWF 파일과 스킨 SWF 파일을 모두 응용 프로그램 서버에 복사해야 합니다. 그러지 않으면 응용 프로그램을 실행할 때 스킨 SWF 파일을 사용할 수 없습니다.

구성 요소 패널에서 FLVPlayback 구성 요소 드래그

1 [구성 요소] 패널에서 더하기(+) 버튼을 클릭하여 비디오 항목을 엽니다.

2 FLVPlayback 구성 요소를 스테이지로 드래그합니다.

3 스테이지에서 FLVPlayback 구성 요소를 선택하고 [구성 요소 관리자]의 [매개 변수] 탭에서 source 매개 변수의 [값] 셀을 찾아 다음 중 하나를 지정하는 문자열을 입력합니다.

- 비디오 파일의 로컬 경로
- 비디오 파일의 URL
- 비디오 파일 재생 방법을 지정하는 SMIL(Syncrhonized Multimedia Integration Language) 파일의 URL

  하나 이상의 FLV 파일을 지정하는 SMIL 파일을 만드는 방법에 대한 자세한 내용은 147페이지의 "SMIL 파일 사용"을 참조하십시오.

4 스테이지에 FLVPlayback 구성 요소가 선택되어 있는 상태로 [구성 요소 관리자]의 [매개 변수] 탭에서 skin 매개 변수에 대한 [값] 셀을 클릭합니다.

5 돋보기 아이콘을 클릭하여 [스킨 선택] 대화 상자를 엽니다.

6 다음 옵션 중 하나를 선택합니다.

- 드롭 다운 [스킨] 목록에서 미리 제작된 스킨 중 하나를 선택하여 구성 요소에 재생 컨트롤 세트를 첨부합니다.
- 사용자 정의 스킨을 만든 경우 팝업 목록에서 [사용자 정의 스킨 URL]을 선택하고, [URL] 상자에 스킨이 들어 있는 SWF 파일의 URL을 입력합니다.
- [없음]을 선택하고 FLV Playback Custom UI 구성 요소를 하나씩 스테이지로 드래그하여 재생 컨트롤을 추가합니다.

  참고: 처음 두 가지 경우 선택한 스킨의 미리 보기가 팝업 메뉴 위의 미리 보기 창에 나타납니다. 색상 선택기를 사용하여 스킨 색상을 변경할 수 있습니다.

  사용자 정의 UI 컨트롤의 색상을 변경하려면 해당 컨트롤을 사용자 정의해야 합니다. 사용자 정의 UI 컨트롤 사용에 대한 자세한 내용은 139페이지의 "개별적인 FLV Playback Custom UI 구성 요소 스키닝"을 참조하십시오.

7 [확인]을 클릭하여 [스킨 선택] 대화 상자를 닫습니다.

8 [컨트롤] > [동영상 테스트]를 선택하여 SWF 파일을 실행하고 비디오를 시작합니다.

다음 절차에서는 [비디오 가져오기] 마법사를 사용하여 FLVPlayback 구성 요소를 추가합니다.

비디오 가져오기 마법사 사용:

1 [파일] > [가져오기] > [비디오 가져오기]를 선택합니다.

2 다음 옵션 중 하나를 선택하여 비디오 파일의 위치를 지정합니다.

- 컴퓨터에서
- 웹 서버, FVSS(Flash Video Streaming Service) 또는 Flash Media Server에 이미 배포됨

3 선택한 파일 위치에 따라 가져오려는 비디오 파일의 위치를 나타내는 경로 또는 URL을 입력한 후 [다음]을 클릭합니다.

4 파일 경로를 선택한 경우 표시된 옵션 중에서 한 가지 비디오 배포 방법을 선택할 수 있는 [배포] 대화 상자가 나타납니다.

- 웹 서버에서 점진적 다운로드
- Flash Video Streaming Service의 스트림
- Flash Media Server의 스트림
- SWF에 비디오를 포함시키고 타임라인에서 재생

  중요: 비디오를 포함시키는 옵션은 선택하지 마십시오. FLVPlayback 구성 요소는 외부 스트리밍 비디오만 재생합니다. 이 옵션은 FLVPlayback 구성 요소를 스테이지에 배치하지 않습니다.

5 [Next]를 클릭합니다.

6 다음 옵션 중 하나를 선택합니다.

- 드롭 다운 [스킨] 목록에서 미리 제작된 스킨 중 하나를 선택하여 구성 요소에 재생 컨트롤 세트를 첨부합니다.
- 구성 요소에 사용할 사용자 정의 스킨을 만든 경우 팝업 목록에서 [사용자 정의 스킨 URL]을 선택하고, [URL] 상자에 해당 스킨이 들어 있는 SWF 파일의 URL을 입력합니다.
- [없음]을 선택하고 FLV Playback Custom UI 구성 요소를 하나씩 스테이지로 드래그하여 재생 컨트롤을 추가합니다.

  참고: 처음 두 가지 경우 선택한 스킨의 미리 보기가 팝업 메뉴 위의 미리 보기 창에 나타납니다.

7 [확인]을 클릭하여 [스킨 선택] 대화 상자를 닫습니다.

8 [비디오 가져오기 완료] 대화 상자에서 다음에 진행될 상황을 읽어 본 후 [완료]를 클릭합니다.

9 FLA 파일을 아직 저장하지 않았다면 [다른 이름으로 저장] 대화 상자가 나타납니다.

10 [컨트롤] > [동영상 테스트]를 선택하여 SWF를 실행하고 비디오를 시작합니다.

다음 절차에서는 ActionScript를 사용하여 FLVPlayback 구성 요소를 추가합니다.

ActionScript를 사용하여 동적으로 인스턴스 만들기:

1 [구성 요소] 패널의 FLVPlayback 구성 요소를 [라이브러리] 패널([윈도우] > [라이브러리])로 드래그합니다.

2 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다. **install_drive**를 Flash를 설치한 드라이브로 변경하고 사용자 설치 환경의 정확한 Skins 폴더 위치를 가리키도록 경로를 수정합니다.

Windows 컴퓨터에서:

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "file:///install_drive|/Program Files/Adobe/Adobe Flash
CS5/en/Configuration/FLVPlayback Skins/ActionScript 3.0/SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
```

Macintosh 컴퓨터에서:

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "file:///Macintosh HD:Applications:Adobe Flash CS5:Configuration:FLVPlayback
Skins:ActionScript 3.0SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
```

참고: source 및 skin 속성을 설정하지 않으면 생성된 동영상 클립이 비어 있는 상태로 나타납니다.

3 [컨트롤] > [동영상 테스트]를 선택하여 SWF 파일을 실행하고 비디오 파일을 시작합니다.

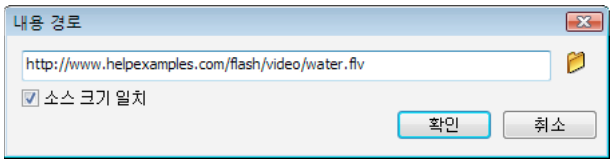
FLVPlayback 구성 요소 매개 변수

[구성 요소 관리자]나 속성 관리자에서 FLVPlayback 구성 요소의 각 인스턴스에 대해 align, autoPlay, cuePoints, preview, scaleMode, skin, skinAutoHide, skinBackgroundAlpha, skinBackgroundColor, source, volume 등의 매개 변수를 설정할 수 있습니다. 이러한 매개 변수에는 각각 같은 이름의 해당 ActionScript 속성이 있습니다. 이러한 매개 변수에 값을 지정하면 그에 따라 응용 프로그램에서 해당 속성의 초기 상태가 설정됩니다. 하지만 ActionScript에서 속성을 설정하면 매개 변수에 지정한 값이 재정의됩니다. 이러한 매개 변수의 가능한 값에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서에서 FLVPlayback 클래스를 참조하십시오.

FLVPlayback source 매개 변수 지정

source 매개 변수를 통해 비디오 파일의 이름과 위치를 지정할 수 있습니다. 이 두 가지 정보는 비디오 파일을 재생하는 방법을 Flash에 알려 줍니다.

[구성 요소 관리자]에서 source 매개 변수의 [값] 셀을 두 번 클릭하여 [내용 경로] 대화 상자를 엽니다.



FLVPlayback 내용 경로 대화 상자

[내용 경로] 대화 상자에는 스테이지에 있는 FLVPlayback 인스턴스가 소스 비디오 파일의 크기와 일치해야 하는지 여부를 지정하는 [소스 FLV 크기 일치] 체크 상자가 있습니다. 소스 비디오 파일에는 재생에 적합한 높이와 폭의 크기 정보가 포함되어 있습니다. 이 옵션을 선택하는 경우 FLVPlayback 인스턴스의 크기가 소스 비디오 파일에 있는 이러한 크기와 일치하도록 조절됩니다.

소스

비디오 파일이나 비디오 파일 재생 방법을 지정하는 XML 파일의 URL 또는 로컬 경로를 입력합니다. 비디오 파일의 정확한 위치를 모르는 경우 폴더 아이콘을 클릭하여 [찾아보기] 대화 상자를 열고 정확한 위치를 찾습니다. 비디오 파일이 대상 SWF 파일이 있는 위치나 그 아래 위치에 있으면 자동으로 SWF 파일 위치에 상대적인 경로가 만들어지므로 웹 서버에서 해당 파일을 찾을 수 있습니다. 이러한 경우가 아니라면 이 경로는 절대 경로(Windows 또는 Macintosh 경로)입니다. 로컬 XML 파일의 이름을 입력하려면 파일의 경로와 이름을 입력합니다.

HTTP URL을 지정하면 비디오 파일은 점진적 다운로드로 재생됩니다. RTMP URL을 지정하면 비디오 파일은 Flash Media Server 또는 FVSS에서 스트리밍됩니다. XML 파일의 URL은 Flash Media Server 또는 FVSS에서 스트리밍되는 비디오 파일일 수도 있습니다.

중요:

다양한 대역폭에서 여러 비디오 파일 스트림을 재생하는 방법을 지정하는 SMIL 파일의 위치를 지정할 수도 있습니다. SMIL(Synchronized Multimedia Integration Language) 파일은 SMIL을 사용하여 FLV 파일을 나타냅니다. SMIL 파일에 대한 설명은 147페이지의 “SMIL 파일 사용”을 참조하십시오.

ActionScript FLVPlayback.source 속성과 FLVPlayback.play() 및 FLVPlayback.load() 메서드를 사용하여 비디오 파일의 이름과 위치를 지정할 수도 있습니다. 이러한 세 가지 대체 방법은 [구성 요소 관리자]의 source 매개 변수보다 우선합니다. 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0 참조 설명서**에서 FLVPlayback 클래스의 FLVPlayback.source, FLVPlayback.play() 및 FLVPlayback.load() 항목을 참조하십시오.

전체 화면 지원

FLVPlayback 구성 요소의 ActionScript 3.0 버전은 전체 화면 모드를 지원하므로 Flash Player 9.0.28.0 이상이 필요하며 전체 화면 보기를 위해서는 HTML이 올바르게 설정되어 있어야 합니다. 미리 제작된 일부 스킨에는 전체 화면 모드를 설정/해제할 수 있는 전환 버튼이 들어 있습니다. 다음 그림에서 FullScreenButton 아이콘은 컨트롤 막대의 오른쪽에 표시됩니다.

컨트롤 막대에 표시된 전체 화면 아이콘

전체 화면 지원은 fullScreenTakeOver 속성이 기본값인 true로 설정된 경우에만 발생합니다.

전체 화면 지원은 하드웨어 가속 지원 여부에 관계없이 발생할 수 있습니다. 하드웨어 가속 지원에 대한 자세한 내용은 129페이지의 “하드웨어 가속”을 참조하십시오.

FLVPlayback의 전체 화면 지원을 구현하려면

1 응용 프로그램에 FLVPlayback 구성 요소를 추가하고 해당 구성 요소에 비디오 파일을 지정합니다.

2 전체 화면 버튼이 있는 FLVPlayback 구성 요소의 스킨(예: SkinUnderPlaySeekFullscreen.swf)을 선택하거나 [구성 요소] 패널의 [비디오] 섹션에서 FLVPlayback 구성 요소에 FullScreenButton 사용자 인터페이스 구성 요소를 추가합니다.

3 [파일] > [제작 설정]을 선택합니다.

4 [제작 설정] 대화 상자에서 [HTML] 탭을 클릭합니다.

5 [HTML] 탭의 [템플릿] 팝업 메뉴에서 [Flash 전용 - 전체 화면 가능]을 선택합니다.

6 또한 [HTML] 탭에서 [Flash 버전 감지] 체크 상자를 선택하고 사용 중인 Flash Player 버전에 따라 9.0.28 이상의 버전을 지정합니다.

7 [포맷] 탭을 선택하고 [Flash(.swf)] 및 [HTML(.html)] 옵션이 모두 선택되어 있는지 확인합니다. 기본 파일 이름을 바꿀 수 있습니다.

8 [제작]을 클릭하고 [확인]을 클릭합니다.

7단계의 대체 방법으로, [확인]을 클릭하고 [파일] > [제작 미리 보기] > [기본값 - (HTML)]을 선택하여 내보낸 HTML 파일을 기본 브라우저에서 자동으로 열 수 있습니다. 또는 브라우저에서 내보낸 HTML 파일을 열어 전체 화면 옵션을 테스트할 수도 있습니다.

웹 페이지에 전체 화면을 지원하는 FLVPlayback 구성 요소를 추가하려면 내보낸 HTML 파일을 열고 SWF 파일을 포함하는 코드를 웹 페이지의 HTML 파일에 복사합니다. 이 코드는 다음 예제와 같습니다.

```
//from the <head> section

<script language="javascript"> AC_FL_RunContent = 0; </script>
<script language="javascript"> DetectFlashVer = 0; </script>
<script src="AC_RunActiveContent.js" language="javascript"></script>
<script language="JavaScript" type="text/javascript">
<!--
// -----------------------------------------------------------------------------
// Globals
// Major version of Flash required
var requiredMajorVersion = 9;
// Minor version of Flash required
var requiredMinorVersion = 0;
// Revision of Flash required
var requiredRevision = 28;
// -----------------------------------------------------------------------------
// -->
</script>

//and from the <body> section

<script language="JavaScript" type="text/javascript">
<!--
if (AC_FL_RunContent == 0 || DetectFlashVer == 0) {
    alert("This page requires AC_RunActiveContent.js.");
} else {
    var hasRightVersion = DetectFlashVer(requiredMajorVersion,
        requiredMinorVersion, requiredRevision);
    if(hasRightVersion) { // if we&apos;ve detected an acceptable version
        // embed the Flash movie
        AC_FL_RunContent(
            &apos;codebase&apos;, &apos;http://download.macromedia.com/pub/
                shockwave/cabs/flash/swflash.cab#version=9,0,28,0&apos;,
            &apos;width&apos;, &apos;550&apos;,
            &apos;height&apos;, &apos;400&apos;,
            &apos;src&apos;, &apos;fullscreen&apos;,
```

```
                &apos;quality&apos;, &apos;high&apos;,
                &apos;pluginspage&apos;, &apos;http://www.macromedia.com/go/
                    getflashplayer&apos;,
                &apos;align&apos;, &apos;middle&apos;,
                &apos;play&apos;, &apos;true&apos;,
                &apos;loop&apos;, &apos;true&apos;,
                &apos;scale&apos;, &apos;showall&apos;,
                &apos;wmode&apos;, &apos;window&apos;,
                &apos;devicefont&apos;, &apos;false&apos;,
                &apos;id&apos;, &apos;fullscreen&apos;,
                &apos;bgcolor&apos;, &apos;#ffffff&apos;,
                &apos;name&apos;, &apos;fullscreen&apos;,
                &apos;menu&apos;, &apos;true&apos;,
                &apos;allowScriptAccess&apos;,&apos;sameDomain&apos;,
                &apos;allowFullScreen&apos;,&apos;true&apos;,
                &apos;movie&apos;, &apos;fullscreen&apos;,
                &apos;salign&apos;, &apos;&apos; ); //end AC code
        } else { // Flash is too old or we can&apos;t detect the plug-in.
            var alternateContent = &apos;Alternative HTML content should be placed
                    here.&apos;
                + &apos;This content requires Adobe Flash Player.&apos;
                + &apos;<a href=http://www.macromedia.com/go/getflash/>Get Flash</a>
                        &apos;;
            document.write(alternateContent); // Insert non-Flash content.
        }
    }
    // -->
    </script>
    <noscript>
        // Provide alternative content for browsers that do not support scripting
        // or for those that have scripting disabled.
        Alternative HTML content should be placed here. This content requires Adobe Flash Player.
        <a href="http://www.macromedia.com/go/getflash/">Get Flash</a>
    </noscript>
```

대체 방법으로, 내보낸 HTML 파일을 웹 페이지의 템플릿으로 사용하고 HTML 파일에 다른 내용을 추가할 수 있습니다. 그러나 이렇게 할 경우에는 나중에 다시 Flash에서 FLVPlayback HTML 파일을 내보낼 때 HTML 파일을 실수로 덮어쓰지 않도록 해당 이름을 변경해야 합니다.

이 경우 HTML 파일과 동일한 폴더로 내보낸 AC_RunActiveContent.js 파일을 웹 서버에 업로드해야 합니다.

전체 화면 모드에 대한 ActionScript 지원에는 fullScreenBackgroundColor, fullScreenSkinDelay, 및 fullScreenTakeOver 속성과 enterFullScreenDisplayState() 메서드가 포함됩니다. 이러한 ActionScript 요소에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서를 참조하십시오.

enterFullScreenDisplayState() 사용

다음 예제와 같이 enterFullScreenDisplayState() ActionScript 메서드를 호출하여 전체 화면 모드를 호출할 수도 있습니다.

```
function handleClick(e:MouseEvent):void {
    myFLVPlybk.enterFullScreenDisplayState();
}
myButton.addEventListener(MouseEvent.CLICK, handleClick);
```

이 예제에서는 FLVPlayback 스킨의 전체 화면 전환 버튼을 클릭하여 전체 화면 모드를 호출하는 것이 **아니라** 웹 페이지의 작성자가 포함시킨 버튼(MyButton)을 클릭하여 전체 화면 모드를 호출합니다. 이 버튼을 클릭하면 handleClick 이벤트 핸들러가 트리거되어 enterFullScreen DisplayState() 메서드가 호출됩니다.

enterFullScreenDisplayState() 메서드는 Stage.displayState 속성을 StageDisplayState.FULL_SCREEN으로 설정하므로 displayState 속성과 동일한 제한이 적용됩니다. enterFullScreenDisplayState() 메서드와 Stage.displayState 속성에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서를 참조하십시오.

하드웨어 가속

Flash Player 9.0.115.0 이상 버전에는 사용 가능한 비디오 하드웨어를 이용하여 FLVPlayback이 전체 화면 모드에서 재생하는 FLV 파일의 성능과 품질을 향상시키는 코드가 포함되어 있습니다. 요구 사항이 충족되고 fullScreenTakeOver 속성이 true로 설정되어 있는 경우 Flash Player에서는 소프트웨어 대신 하드웨어 가속을 사용하여 비디오 파일의 크기를 조절합니다. FLVPlayback 구성 요소가 이전 버전의 Flash Player에서 실행되거나 하드웨어 가속을 위한 요구 사항이 충족되지 않는 경우 Flash Player에서는 이전의 방법대로 비디오 파일의 크기를 확대합니다.

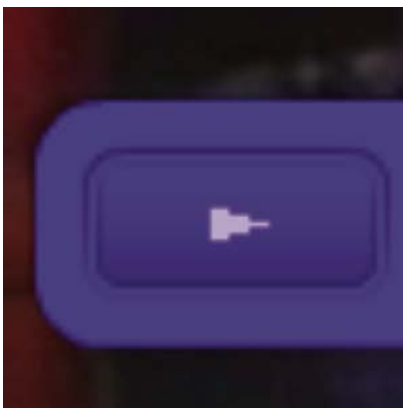
전체 화면 지원에 하드웨어 가속을 이용하려면 컴퓨터에 4MB 이상의 VRAM(비디오 RAM)이 장착된 DirectX 7 호환 비디오 카드가 있어야 합니다. 이 하드웨어 지원은 Windows 2000 또는 Mac OS X 10.2 이상의 운영 체제에서 사용할 수 있습니다. Direct X®에서는 소프트웨어와 비디오 하드웨어 간의 인터페이스를 구성하여 3차원 및 2차원 그래픽 등의 속도를 높여 주는 API를 제공합니다.

하드웨어 가속 모드를 이용하려면 다음 중 한 가지 방법으로 전체 화면 모드를 호출해야 합니다.

- FLVPlayback 스킨의 전체 화면 전환 버튼 사용
- FullScreenButton 비디오 컨트롤 사용
- ActionScript enterFullScreenDisplayState() 메서드 사용. 자세한 내용은 128페이지의 “enterFullScreenDisplayState() 사용”을 참조하십시오.

  Stage.displayState 속성을 StageDisplayState.FULLSCREEN으로 설정하여 전체 화면 모드를 호출하면 FLVPlayback에서는 비디오 하드웨어 및 메모리를 사용할 수 있는 경우에도 하드웨어 가속을 사용하지 않습니다.

  전체 화면 지원을 위해 하드웨어 가속을 사용할 경우 FLVPlayback 스킨의 크기가 비디오 플레이어 및 비디오 파일과 함께 조절됩니다. 다음 그림에서는 FLVPlayback 스킨에 하드웨어 가속을 사용한 전체 화면 모드의 결과를 최대 해상도로 자세히 보여 줍니다.



320x240 픽셀 비디오를 사용한 1600x1200 모니터의 전체 화면 모드

이 그림에서는 1600 x 1200 모니터에서 기본 FLVPlayback 크기인 폭 320, 높이 240의 비디오 파일을 전체 화면 모드를 사용하여 재생한 결과를 보여 줍니다. 스킨의 왜곡 효과는 FLV 파일의 크기가 작거나 모니터 크기가 클수록 뚜렷이 나타나며, FLV 파일의 크기가 크거나 모니터 크기가 작을수록 적게 나타납니다. 예를 들어, 모니터를 640 x 480에서 1600 x 1200으로 변경하면 스킨의 크기도 늘어나지만 왜곡 효과는 적게 나타납니다.

skinScaleMaximum 속성을 설정하여 FLVPlayback 스킨의 크기 조절을 제한할 수 있습니다. 기본값은 4.0 또는 400%입니다. 그러나 스킨의 크기 조절을 제한하면 하드웨어와 소프트웨어를 함께 사용해야 FLV의 크기를 조절할 수 있으며 따라서 높은 비트율에서 인코딩된 큰 크기의 FLV에서는 성능이 저하될 수 있습니다. 비디오가 큰 경우(예를 들어, 폭 640픽셀 이상, 높이 480픽셀 이상) skinScaleMaximum을 작은 값으로 설정하지 마십시오. 그러면 대형 디스플레이 모니터에서 현저한 성능 문제가 발생할 수 있습니다. skinScaleMaximum 속성을 사용하면 성능과 품질을 모두 적절한 수준으로 관리하고 큰 스킨의 모양을 관리할 수 있습니다.

전체 화면 모드 종료

전체 화면 모드를 종료하려면 전체 화면 버튼을 다시 클릭하거나 Esc 키를 누릅니다.

height, registrationHeight, registrationWidth, registrationX, registrationY, scaleX, scaleY, width, x 및 y 속성을 설정하고 setScale() 또는 setSize() 메서드를 호출하면 레이아웃이 변경되어 FLVPLayback 구성 요소의 전체 화면 모드가 종료될 수 있습니다.

align 또는 scaleMode 속성을 설정할 경우 FLVPlayback에서는 전체 화면 모드가 종료될 때까지 해당 속성을 center 및 maintainAspectRatio로 설정합니다.

전체 화면을 사용하고 있을 때 fullScreenTakeOver 속성 값을 true에서 false로 변경하면 Flash에서 하드웨어 가속 모드로 전환되어 전체 화면 모드가 종료됩니다.

여러 비디오 파일 재생을 위한 레이아웃 정렬

ActionScript 3.0 FLVPlayback에는 비디오 파일의 크기를 조절할 때 비디오 파일을 중앙에 배치할지 아니면 구성 요소의 위쪽, 아래쪽, 왼쪽 또는 오른쪽에 배치할지 여부를 지정하는 align 속성이 있습니다. ActionScript 3.0 구성 요소에는 구성 요소의 x, y, width 및 height 속성 이외에도 registrationX, registrationY, registrationWidth 및 registrationHeight 속성이 있습니다. 이러한 속성은 처음에는 x, y, width 및 height 속성과 일치합니다. 이후에 비디오 파일을 로드하면 자동으로 레이아웃을 재구성해도 비디오 파일이 변경되지 않으므로 새 비디오 파일을 같은 위치의 중앙에 배치할 수 있습니다. scaleMode = VideoScaleMode.MAINTAIN_ASPECT_RATIO일 경우 이후 FLV 파일은 구성 요소의 폭과 높이를 변경하지 않고 구성 요소의 원래 크기에 맞춰질 수 있습니다.

점진적으로 다운로드되는 비디오 파일 자동 재생

점진적으로 다운로드되는 비디오 파일을 로드할 경우 비디오 파일이 처음부터 끝까지 재생될 수 있도록 충분히 다운로드된 경우에만 FLVPlayback에서 비디오 파일 재생을 시작합니다.

충분히 다운로드되기 전에 비디오 파일을 재생하려면 매개 변수 없이 play() 메서드를 호출하십시오.

비디오 파일이 충분히 다운로드될 때까지 다시 기다리려면 pause() 메서드를 호출하고 playWhenEnoughDownloaded() 메서드를 호출하십시오.

큐 포인트 사용

큐 포인트는 비디오 파일이 재생되는 동안 비디오 플레이어에서 cuePoint 이벤트를 전달하는 포인트입니다. 웹 페이지의 다른 요소와 상호 작용하려는 시점마다 FLV 파일에 큐 포인트를 추가할 수 있습니다. 예를 들어, 텍스트 또는 그래픽을 표시하거나, Flash 애니메이션과 동기화하거나, FLV 파일을 일시 정지하여 재생에 영향을 주거나, 비디오의 다른 포인트를 검색하거나, 다른 FLV 파일로 전환하는 등의 작업에 큐 포인트를 사용할 수 있습니다. 큐 포인트를 사용하면 ActionScript 코드에서 제어를 받을 수 있으며 FLV 파일의 포인트를 웹 페이지 상의 다른 액션과 동기화할 수 있습니다.

큐 포인트에는 내비게이션, 이벤트 및 ActionScript의 세 가지 유형이 있습니다. 내비게이션 및 이벤트 큐 포인트는 FLV 파일 스트림 및 FLV 파일의 메타데이터 패킷에 포함되기 때문에 **포함된** 큐 포인트라고도 합니다.

내비게이션 큐 포인트는 FLV 파일 내에서 사용자가 지정한 시간에 가장 근접한 지점에 키 프레임을 만들기 때문에 이 큐 포인트를 사용하면 FLV 파일의 특정 프레임을 검색할 수 있습니다. **키 프레임**은 FLV 파일 스트림의 이미지 프레임 사이에 있는 데이터 세그먼트입니다. 내비게이션 큐 포인트를 검색하면 구성 요소에서 해당 키 프레임을 검색하고 cuePoint 이벤트를 시작합니다.

이벤트 큐 포인트를 사용하면 FLV 파일 내에서 특정 포인트를 웹 페이지의 외부 이벤트에 맞춰 동기화할 수 있습니다. cuePoint 이벤트는 지정된 시간에 정확하게 발생합니다. 비디오 가져오기 마법사나 Flash Video Encoder를 사용하여 내비게이션 및 이벤트 큐 포인트를 FLV 파일에 포함할 수 있습니다. 비디오 가져오기 마법사 및 Flash Video Encoder에 대한 자세한 내용은 **Flash 사용 설명서**의 16장, "비디오를 사용한 작업"을 참조하십시오.

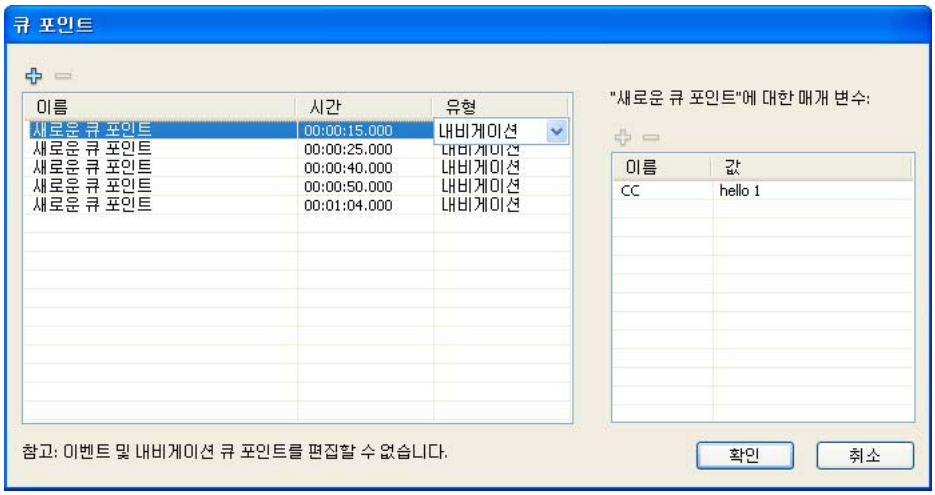
ActionScript 큐 포인트는 구성 요소의 [Flash 비디오 큐 포인트] 대화 상자 또는 FLVPlayback.addASCuePoint() 메서드를 통해 추가할 수 있는 외부 큐 포인트입니다. 구성 요소는 FLV 파일과 별도로 ActionScript 큐 포인트를 저장하고 추적합니다. 그러므로 이 큐 포인트는 포함된 큐 포인트보다 정확도가 떨어집니다. ActionScript 큐 포인트의 정확도는 10분의 1초입니다. 재생 헤드가 업데이트될 때 구성 요소에서는 ActionScript 큐 포인트에 대한 cuePoint 이벤트를 생성하기 때문에 playheadUpdateInterval 속성의 값을 낮게 설정하면 ActionScript 큐 포인트의 정확도를 높일 수 있습니다. 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서에서 FLVPlayback.playheadUpdateInterval 속성을 참조하십시오.

ActionScript 및 FLV 파일의 메타데이터에서 큐 포인트는 name, time, type 및 parameters 속성이 있는 객체로 표현됩니다. name 속성은 큐 포인트의 이름을 포함하는 문자열입니다. time 속성은 큐 포인트가 발생하는 시간을 시, 분, 초, 밀리초(HH:MM:SS.mmm)로 나타내는 숫자입니다. type 속성은 만든 큐 포인트의 유형에 따라 "navigation", "event" 또는 "actionscript"라는 값을 갖는 문자열입니다. parameters 속성은 지정된 이름과 값의 쌍으로 이루어진 배열입니다.

cuePoint 이벤트가 발생할 때 info 속성을 통해 이벤트 객체에서 큐 포인트 객체를 사용할 수 있습니다.

Flash 비디오 큐 포인트 대화 상자 사용

[구성 요소 관리자]에서 cuePoints 매개 변수의 [값] 셀을 두 번 클릭하면 [Flash 비디오 큐 포인트] 대화 상자가 열립니다. 이 대화 상자는 다음 그림과 같이 나타납니다.



큐 포인트 대화 상자

이 대화 상자는 포함된 큐 포인트와 ActionScript 큐 포인트를 표시합니다. 이 대화 상자를 사용하여 ActionScript 큐 포인트 및 큐 포인트 매개 변수를 추가하거나 삭제할 수 있습니다. 또한 포함된 큐 포인트를 활성화하거나 비활성화할 수 있습니다. 그러나 포함된 큐 포인트를 추가, 변경 또는 삭제할 수는 없습니다.

ActionScript 큐 포인트 추가:

1. [구성 요소 관리자]에서 cuePoints 매개 변수의 [값] 셀을 두 번 클릭하여 [Flash 비디오 큐 포인트] 대화 상자를 엽니다.
2. 왼쪽 위 모서리(큐 포인트 목록 위)에 있는 더하기(+) 기호를 클릭하여 기본 ActionScript 큐 포인트 항목을 추가합니다.
3. [이름] 열에서 [새로운 큐 포인트] 텍스트를 클릭하고 해당 큐 포인트에 지정할 이름으로 편집합니다.
4. [시간] 열의 값 00:00:00:000을 편집할 수 있도록 클릭하고 해당 큐 포인트가 발생할 시간을 지정합니다. 시간은 시, 분, 초, 밀리초(HH:MM:SS.mmm) 형식으로 지정할 수 있습니다.

   큐 포인트가 여러 개 있는 경우 새로운 큐 포인트는 목록에서 시간 순에 따라 해당 위치로 이동됩니다.
5. 선택한 큐 포인트에 대한 매개 변수를 추가하려면 [매개 변수] 섹션 위에 있는 더하기(+) 기호를 클릭하고 [이름] 및 [값] 열에 필요한 값을 입력합니다. 각 매개 변수에 대해 이 단계를 반복합니다.
6. ActionScript 큐 포인트를 계속 추가하려면 각 큐 포인트에 대해 2~5단계를 반복합니다.

7 [확인]을 클릭하여 변경 내용을 저장합니다.

ActionScript 큐 포인트 삭제:

1 [구성 요소 관리자]에서 cuePoints 매개 변수의 [값] 셀을 두 번 클릭하여 [Flash 비디오 큐 포인트] 대화 상자를 엽니다.

2 삭제하려는 큐 포인트를 선택합니다.

3 왼쪽 위 모서리(큐 포인트 목록 위)에 있는 빼기(-) 기호를 클릭하여 선택한 큐 포인트를 삭제합니다.

4 삭제하려는 각 큐 포인트에 대해 2~3단계를 반복합니다.

5 [확인]을 클릭하여 변경 내용을 저장합니다.

포함된 FLV 파일 큐 포인트를 활성화 또는 비활성화하려면

1 [구성 요소 관리자]에서 cuePoints 매개 변수의 [값] 셀을 두 번 클릭하여 [Flash 비디오 큐 포인트] 대화 상자를 엽니다.

2 활성화 또는 비활성화할 큐 포인트를 선택합니다.

3 [유형] 열의 값을 클릭하여 팝업 메뉴를 트리거하거나 아래쪽 화살표를 클릭합니다.

4 활성화할 큐 포인트 유형 이름(예: [이벤트] 또는 [내비게이션])을 클릭합니다. [비활성화]를 클릭하여 해당 큐 포인트를 비활성화합니다.

5 [확인]을 클릭하여 변경 내용을 저장합니다.

ActionScript를 통해 큐 포인트 사용

ActionScript를 사용하여 ActionScript 큐 포인트 추가, cuePoint 이벤트 수신, 모든 유형 또는 특정 유형의 큐 포인트 찾기, 내비게이션 큐 포인트 검색, 큐 포인트 활성화/비활성화, 큐 포인트의 활성화 여부 확인, 큐 포인트 제거 등의 다양한 작업을 수행할 수 있습니다.

이 단원의 예제에서는 다음과 같은 세 가지 큐 포인트가 포함된 cuepoints.flv라는 FLV 파일을 사용합니다.

이름	시간	유형
point1	00:00:00.418	내비게이션
point2	00:00:07.748	내비게이션
point3	00:00:16.020	내비게이션

ActionScript 큐 포인트 추가

addASCuePoint() 메서드를 사용하여 FLV 파일에 ActionScript 큐 포인트를 추가할 수 있습니다. 다음 예제에서는 FLV 파일이 재생 준비가 되었을 때 두 개의 ActionScript 큐 포인트를 FLV 파일에 추가합니다. 첫 번째 큐 포인트는 큐 포인트의 시간, 이름 및 유형을 속성으로 지정하는 큐 포인트 객체를 사용하여 추가합니다. 두 번째 호출에서는 메서드의 time 및 name 매개 변수를 사용하여 시간과 이름을 지정합니다.

```
// Requires an FLVPlayback instance called my_FLVPlybk on Stage
import fl.video.*;
import fl.video.MetadataEvent;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
var cuePt:Object = new Object(); //create cue point object
cuePt.time = 2.02;
cuePt.name = "ASpt1";
cuePt.type = "actionscript";
my_FLVPlybk.addASCuePoint(cuePt);//add AS cue point
// add 2nd AS cue point using time and name parameters
my_FLVPlybk.addASCuePoint(5, "ASpt2");
```

자세한 내용은 **Adobe Flash Professional CS5**용 ActionScript 3.0 참조 설명서에서 FLVPlayback.addASCuePoint() 메서드를 참조하십시오.

cuePoint 이벤트 수신

cuePoint 이벤트를 사용하면 cuePoint 이벤트가 발생할 때 ActionScript 코드에서 제어를 받을 수 있습니다. 다음 예제에서 큐 포인트가 발생하면 cuePoint 리스너는 playheadTime 속성 및 해당 큐 포인트의 이름과 유형을 표시하는 이벤트 핸들러 함수를 호출합니다. 결과를 보려면 이 예제와 이전 단원에 있는 ActionScript 큐 포인트 추가 예제를 함께 사용하십시오.

```
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
        trace("Elapsed time in seconds: " + my_FLVPlybk.playheadTime);
        trace("Cue point name is: " + eventObject.info.name);
        trace("Cue point type is: " + eventObject.info.type);
}
```

cuePoint 이벤트에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서에서 FLVPlayback.cuePoint 이벤트를 참조하십시오.

큐 포인트 찾기

ActionScript를 사용하여 모든 유형의 큐 포인트를 찾거나 지정 시간에 가장 근접한 큐 포인트를 찾거나 특정 이름을 갖는 다음 큐 포인트를 찾을 수 있습니다.

다음 예제에서 ready_listener() 이벤트 핸들러는 findCuePoint() 메서드를 호출하여 ASpt1 큐 포인트를 찾은 다음 findNearestCuePoint() 메서드를 호출하여 ASpt1 큐 포인트의 시간에 가장 근접한 내비게이션 큐 포인트를 찾습니다.

```
import fl.video.FLVPlayback;
import fl.video.CuePointType;
import fl.video.VideoEvent;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
var rtn_obj:Object; //create cue point object
my_FLVPlybk.addASCuePoint(2.02, "ASpt1");//add AS cue point
function ready_listener(eventObject:VideoEvent):void {
    rtn_obj = my_FLVPlybk.findCuePoint("ASpt1", CuePointType.ACTIONSCRIPT);
    traceit(rtn_obj);
    rtn_obj = my_FLVPlybk.findNearestCuePoint(rtn_obj.time, CuePointType.NAVIGATION);
    traceit(rtn_obj);
}
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function traceit(cuePoint:Object):void {
    trace("Cue point name is: " + cuePoint.name);
    trace("Cue point time is: " + cuePoint.time);
    trace("Cue point type is: " + cuePoint.type);
}
```

다음 예제에서는 ready_listener() 이벤트 핸들러가 ASpt 큐 포인트를 찾고 findNextCuePointWithName() 메서드를 호출하여 동일한 이름의 다음 큐 포인트를 찾습니다.

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
var rtn_obj:Object; //create cue point object
my_FLVPlybk.addASCuePoint(2.02, "ASpt");//add AS cue point
my_FLVPlybk.addASCuePoint(3.4, "ASpt");//add 2nd Aspt
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
    rtn_obj = my_FLVPlybk.findCuePoint("ASpt", CuePointType.ACTIONSCRIPT);
    traceit(rtn_obj);
    rtn_obj = my_FLVPlybk.findNextCuePointWithName(rtn_obj);
    traceit(rtn_obj);
}
function traceit(cuePoint:Object):void {
    trace("Cue point name is: " + cuePoint.name);
    trace("Cue point time is: " + cuePoint.time);
    trace("Cue point type is: " + cuePoint.type);
}
```

큐 포인트를 찾는 방법에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 ActionScript 3.0 참조 설명서에서 FLVPlayback.findCuePoint(), FLVPlayback.findNearestCuePoint() 및 FLVPlayback.findNextCuePointWithName() 메서드를 참조하십시오.

내비게이션 큐 포인트 검색

내비게이션 큐 포인트 검색, 지정한 시간 이후의 다음 내비게이션 큐 포인트 검색 및 지정한 시간 앞의 이전 내비게이션 검색 등의 작업을 수행할 수 있습니다. 다음 예제에서는 FLV 파일 cuepoints.flv를 재생하고 ready 이벤트가 발생할 때 7.748 위치의 큐 포인트를 검색합니다. cuePoint 이벤트가 발생할 때 이 예제에서는 seekToPrevNavCuePoint() 메서드를 호출하여 첫 번째 큐 포인트를 검색합니다. 해당 cuePoint 이벤트가 발생할 때 이 예제에서는 seekToNextNavCuePoint() 메서드를 호출하여 eventObject.info.time(현재 큐 포인트의 시간)에 10초를 추가하는 방식으로 마지막 큐 포인트를 검색합니다.

```
import fl.video.*;

my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:Object):void {
    my_FLVPlybk.seekToNavCuePoint("point2");
}
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
    trace(eventObject.info.time);
    if(eventObject.info.time == 7.748)
        my_FLVPlybk.seekToPrevNavCuePoint(eventObject.info.time - .005);
    else
        my_FLVPlybk.seekToNextNavCuePoint(eventObject.info.time + 10);
}
my_FLVPlybk.source = "http://helpexamples.com/flash/video/cuepoints.flv";
```

자세한 내용은 **Adobe Flash Professional CS5**용 ActionScript 3.0 참조 설명서에서 FLVPlayback.seekToNavCuePoint(), FLVPlayback.seekToNextNavCuePoint() 및 FLVPlayback.seekToPrevNavCuePoint() 메서드를 참조하십시오.

포함된 FLV 파일 큐 포인트 활성화 및 비활성화

setFLVCuePointEnabled() 메서드를 사용하여 포함된 FLV 파일 큐 포인트를 활성화 및 비활성화할 수 있습니다. 비활성화된 큐 포인트는 cuePoint 이벤트를 트리거하지 않으며 seekToCuePoint(), seekToNextNavCuePoint() 또는 seekToPrevNavCuePoint() 메서드에 반응하지 않습니다. 그렇지만 비활성화된 큐 포인트도 findCuePoint(), findNearestCuePoint() 및 findNextCuePointWithName() 메서드를 사용하여 찾을 수는 있습니다.

isFLVCuePointEnabled() 메서드를 사용하여 포함된 FLV 파일 큐 포인트가 활성화되었는지 확인할 수 있습니다. 다음 예제에서는 비디오 재생 준비가 되었을 때 포함된 큐 포인트 point2 및 point3을 비활성화합니다. 하지만, 첫 번째 cuePoint 이벤트가 발생할 때 이벤트 핸들러에서는 point3 큐 포인트가 비활성화되었는지 확인하고 비활성화된 경우 이를 활성화합니다.

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv";
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
    my_FLVPlybk.setFLVCuePointEnabled(false, "point2");
    my_FLVPlybk.setFLVCuePointEnabled(false, "point3");
}
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
        trace("Cue point time is: " + eventObject.info.time);
        trace("Cue point name is: " + eventObject.info.name);
        trace("Cue point type is: " + eventObject.info.type);
        if (my_FLVPlybk.isFLVCuePointEnabled("point2") == false) {
            my_FLVPlybk.setFLVCuePointEnabled(true, "point2");
        }
}
```

자세한 내용은 **Adobe Flash Professional CS5**용 ActionScript 3.0 참조 설명서에서 FLVPlayback.isFLVCuePointEnabled() 및 FLVPlayback.setFLVCuePointEnabled() 메서드를 참조하십시오.

ActionScript 큐 포인트 제거

removeASCuePoint() 메서드를 사용하여 ActionScript 큐 포인트를 제거할 수 있습니다. 다음 예제에서는 ASpt1 큐 포인트가 발생할 때 ASpt2 큐 포인트를 제거합니다.

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
my_FLVPlybk.addASCuePoint(2.02, "ASpt1");//add AS cue point
my_FLVPlybk.addASCuePoint(3.4, "ASpt2");//add 2nd Aspt
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
        trace("Cue point name is: " + eventObject.info.name);
        if (eventObject.info.name == "ASpt1") {
            my_FLVPlybk.removeASCuePoint("ASpt2");
            trace("Removed cue point ASpt2");
        }
}
```

자세한 내용은 **Adobe Flash Professional CS5**용 ActionScript 3.0 참조 설명서에서 FLVPlayback.removeASCuePoint()를 참조하십시오.

여러 비디오 파일 재생

이전 비디오 파일의 재생이 끝날 때 source 속성에 새로운 URL을 로드하는 간단한 방법으로 FLVPlayback 인스턴스에서 여러 비디오 파일을 순차적으로 재생할 수 있습니다. 예를 들어, 다음 ActionScript 코드는 비디오 파일의 재생이 끝날 때 발생하는 complete 이벤트를 수신합니다. 이 이벤트가 발생하면 코드에서는 source 속성에 새로운 비디오 파일의 이름과 위치를 설정하고 play() 메서드를 호출하여 새 비디오를 재생합니다.

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/clouds.flv";
my_FLVPlybk.addEventListener(VideoEvent.COMPLETE, complete_listener);
// listen for complete event; play new FLV
function complete_listener(eventObject:VideoEvent):void {
    if (my_FLVPlybk.source == "http://www.helpexamples.com/flash/video/clouds.flv") {
        my_FLVPlybk.play("http://www.helpexamples.com/flash/video/water.flv");
    }
};
```

여러 비디오 플레이어 사용

하나의 FLVPlayback 구성 요소 인스턴스 내에서 비디오 플레이어를 여러 개 열어 다수의 비디오를 재생하고 재생 중인 비디오 사이를 전환할 수 있습니다.

FLVPlayback 구성 요소를 스테이지로 드래그하면 첫 번째 비디오 플레이어가 만들어집니다. 구성 요소에서는 처음에 만들어진 비디오 플레이어에 숫자 0을 지정하고 이를 기본 플레이어로 설정합니다. 추가 비디오 플레이어를 만들려면 activeVideoPlayerIndex 속성에 새 값을 설정합니다. activeVideoPlayerIndex 속성을 설정하면 지정한 비디오 플레이어가 **활성** 비디오 플레이어가 되고 FLVPlayback 클래스의 속성과 메서드에 따라 영향을 받게 됩니다. activeVideoPlayerIndex 속성을 설정한다고 해서 해당 비디오 플레이어가 표시되는 것은 아닙니다. 비디오 플레이어가 표시되게 하려면 visibleVideoPlayerIndex 속성을 해당 비디오 플레이어의 번호로 설정합니다. 이러한 속성이 FLVPlayback 클래스의 메서드 및 속성과 상호 작용하는 방법에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 ActionScript 3.0 참조 설명서에서 FLVPlayback.activeVideoPlayerIndex 및 FLVPlayback.visibleVideoPlayerIndex 속성을 참조하십시오.

다음 ActionScript 코드는 source 속성을 로드하여 기본 비디오 플레이어에서 비디오 파일을 재생하고 비디오 파일에 큐 포인트를 추가합니다. ready 이벤트가 발생하면 이벤트 핸들러에서는 activeVideoPlayerIndex 속성을 숫자 1로 설정하여 두 번째 비디오 플레이어를 엽니다. 두 번째 비디오 플레이어에 대한 FLV 파일과 큐 포인트를 지정한 다음 기본 플레이어(0)를 다시 활성 비디오 플레이어로 설정합니다.

```
/**
    Requires:
- FLVPlayback component on the Stage with an instance name of my_FLVPlybk
*/
// add a cue point to the default player
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/clouds.flv";
my_FLVPlybk.addASCuePoint(3, "1st_switch");
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
    // add a second video player and create a cue point for it
    my_FLVPlybk.activeVideoPlayerIndex = 1;
    my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
    my_FLVPlybk.addASCuePoint(3, "2nd_switch");
    my_FLVPlybk.activeVideoPlayerIndex = 0;
};
```

FLV 파일이 재생되고 있는 동안 다른 FLV 파일로 전환하려면 ActionScript 코드에 전환을 만들어야 합니다. 큐 포인트를 사용하면 cuePoint 이벤트를 통해 FLV 파일의 특정 포인트에서 필요한 작업을 수행할 수 있습니다. 다음 코드는 cuePoint 이벤트에 대한 리스너를 만든 다음, 활성 비디오 플레이어(0)를 일시 정지하고 두 번째 플레이어(1)로 전환하여 해당 FLV 파일을 재생하는 핸들러 함수를 호출합니다.

```
import fl.video.*;
// add listener for a cuePoint event
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
// add the handler function for the cuePoint event
function cp_listener(eventObject:MetadataEvent):void {
        // display the no. of the video player causing the event
        trace("Hit cuePoint event for player: " + eventObject.vp);
        // test for the video player and switch FLV files accordingly
        if (eventObject.vp == 0) {
            my_FLVPlybk.pause(); //pause the first FLV file
            my_FLVPlybk.activeVideoPlayerIndex = 1; // make the 2nd player active
            my_FLVPlybk.visibleVideoPlayerIndex = 1; // make the 2nd player visible
            my_FLVPlybk.play(); // begin playing the new player/FLV
        } else if (eventObject.vp == 1) {
            my_FLVPlybk.pause(); // pause the 2nd FLV
            my_FLVPlybk.activeVideoPlayerIndex = 0; // make the 1st player active
            my_FLVPlybk.visibleVideoPlayerIndex = 0; // make the 1st player visible
            my_FLVPlybk.play(); // begin playing the 1st player
        }
}
my_FLVPlybk.addEventListener(VideoEvent.COMPLETE, complete_listener);
function complete_listener(eventObject:VideoEvent):void {
        trace("Hit complete event for player: " + eventObject.vp);
        if (eventObject.vp == 0) {
            my_FLVPlybk.activeVideoPlayerIndex = 1;
            my_FLVPlybk.visibleVideoPlayerIndex = 1;
            my_FLVPlybk.play();
        } else {
            my_FLVPlybk.closeVideoPlayer(1);
        }
};
```

새 비디오 플레이어를 만들면 FLVPlayback 인스턴스는 해당 속성을 기본 비디오 플레이어의 값으로 설정합니다. 단, source, totalTime 및 isLive 속성은 예외로서, FLVPlayback 인스턴스에서는 이러한 속성을 항상 기본값인 빈 문자열, 0 및 false로 각각 설정합니다. 또한 기본 비디오 플레이어에 대해 기본값 true로 설정되는 autoPlay 속성을 false로 설정합니다. cuePoints 속성은 아무런 영향을 주지 않으며 기본 비디오 플레이어의 후속 로드 작업에도 영향을 주지 않습니다.

볼륨, 위치, 크기, 가시성 및 사용자 인터페이스 컨트롤을 제어하는 메서드 및 속성은 항상 전역 메서드 및 속성이며 activeVideoPlayerIndex 속성 설정 값에 따라 해당 비헤이비어가 영향을 받지 않습니다. 이러한 메서드 및 속성과 activeVideoPlayerIndex 속성 설정의 효과에 대한 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0 참조 설명서**에서 FLVPlayback.activeVideoPlayerIndex 속성을 참조하십시오. 나머지 속성 및 메서드는 activeVideoPlayerIndex 속성 값으로 식별되는 비디오 플레이어를 대상으로 합니다.

그러나 크기를 제어하는 속성 및 메서드는 visibleVideoPlayerIndex 속성에 따라**영향을 받습니다**. 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0 참조 설명서**에서 FLVPlayback.visibleVideoPlayerIndex 속성을 참조하십시오.

Flash Media Server에서 FLV 파일 스트리밍

Flash Media Server에서 FLV 파일을 스트리밍하기 위한 요구 사항은 Flash Video Streaming Service 공급자가 기본 대역폭 탐지 기능을 제공하는지 여부에 따라 다릅니다. 기본 대역폭 탐지 기능이 제공되면 대역폭 탐지가 스트리밍 서버에 내장되어 있으므로 성능이 향상됩니다. 기본 대역폭 탐지 기능이 사용 가능한지 여부는 공급자에게 문의하십시오.

Flash Media Server에 있는 FLV 파일에 액세스하려면 rtmp://**my_servername/my_application/stream.**flv와 같은 URL을 사용합니다.

Flash Media Server에서 라이브 스트림을 재생할 때는 FLVPlayback의 isLive 속성을 true로 설정해야 합니다. 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0 참조 설명서**에서 FLVPlayback.isLive 속성을 참조하십시오.

라이브 스트림 설정 방법을 포함하여 Flash Media Server 관리에 대한 자세한 내용은 www.adobe.com/support/documentation/kr/flashmediaserver/의 Flash Media Server 설명서를 참조하십시오.

기본 대역폭 탐지를 사용하거나 대역폭 탐지 기능이 없는 경우

NCManagerNative 클래스는 기본 대역폭 탐지를 지원하는 NCManager의 하위 클래스입니다. 기본 대역폭 탐지는 일부 Flash Video Streaming Service 공급자에서만 지원할 수 있습니다. NCManagerNative를 사용할 경우 Flash Media Server에서 특별한 파일이 필요하지 않습니다. NCManagerNative를 사용하면 대역폭 탐지가 필요하지 않을 경우 main.asc 파일 없이도 모든 Flash Media Server 버전에 연결할 수 있습니다.

기본 NCManager 클래스 대신 NCManagerNative를 사용하려면 FLA 파일의 첫 번째 프레임에 다음 코드 행을 추가하십시오.

```
import fl.video*;
VideoPlayer.iNCManagerClass = fl.video.NCManagerNative;
```

기본 대역폭 탐지 이외의 대역폭 탐지 기능을 사용할 경우

Flash Video Streaming Service 공급자가 기본 대역폭 탐지를 제공하지 않지만 대역폭 탐지를 사용해야 하는 경우 main.asc 파일을 Flash Media Server FLV 응용 프로그램에 추가해야 합니다. www.adobe.com/go/learn_fl_samples_kr에서 온라인으로 main.asc 파일을 찾을 수 있습니다. 이 파일은 Samples\ComponentsAS2\FLVPlayback 디렉토리 내의 Samples.zip 파일에 들어 있습니다.

FLV 파일 스트리밍을 위해 Flash Media Server를 설정하려면

1. 사용자의 Flash Media Server 응용 프로그램 폴더에 새로운 폴더를 하나 만들고 **my_application**과 같은 이름을 지정합니다.
2. main.asc 파일을 방금 만든 my_application 폴더에 복사합니다.
3. my_application 폴더에 **streams**라는 이름으로 새 폴더를 만듭니다.
4. streams 폴더 내에 **_definst_**라는 이름으로 새 폴더를 만듭니다.

5 사용하려는 FLV 파일을 **_definst_** 폴더에 배치합니다.

FLVPlayback 구성 요소 사용자 정의

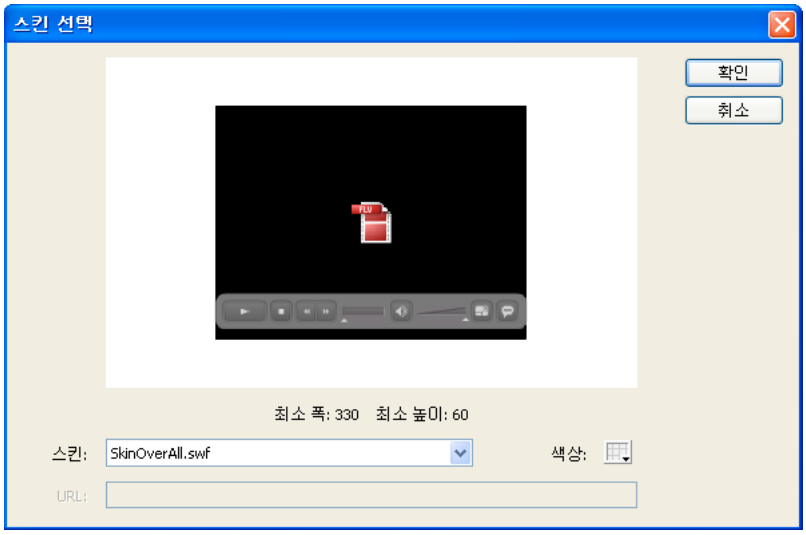
이 단원에서는 FLVPlayback 구성 요소를 사용자 정의하는 방법에 대해 설명합니다. 다른 구성 요소를 사용자 정의할 때 사용하는 대부분의 방법은 FLVPlayback 구성 요소에서 사용할 수 없습니다. FLVPlayback 구성 요소를 사용자 정의하려면 이 단원에서 설명하는 방법만 사용하십시오.

FLVPlayback 구성 요소를 사용자 정의하는 경우 미리 제작된 스킨을 선택하거나 FLV Playback Custom UI 구성 요소를 개별적으로 스키닝하거나 새 스킨을 작성하는 세 가지 방법을 사용할 수 있습니다. FLVPlayback 속성을 사용하여 스킨의 비헤이비어를 수정할 수도 있습니다.

참고: FLVPlayback 구성 요소에 스킨을 사용하려면 응용 프로그램 SWF 파일과 함께 해당 스킨 SWF 파일을 웹 서버에 업로드해야 합니다.

미리 제작된 스킨 선택

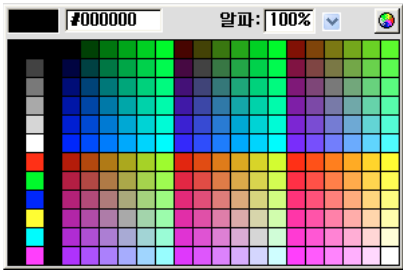
FLVPlayback 구성 요소에 대한 스킨을 선택하려면 [구성 요소 관리자]에서 skin 매개 변수에 대한 값 셀을 클릭한 다음 돋보기 아이콘을 클릭하여 다음과 같은 [스킨 선택] 대화 상자를 열고 이 대화 상자에서 사용할 스킨을 선택하거나 스킨 SWF 파일 위치를 지정하는 URL을 입력합니다.



FLVPlayback 스킨 선택 대화 상자

[스킨] 팝업 메뉴에 나열되는 스킨은 Flash 응용 프로그램 폴더 /Flash Configuration/FLVPlayback Skins/ActionScript 3.0에 있습니다. 새 스킨을 만들고 해당 SWF 파일을 이 폴더에 배치하면 새 스킨을 이 대화 상자에서 사용할 수 있습니다. 팝업 메뉴에 나타나는 스킨 이름은 .swf 확장명을 갖고 있습니다. 스킨 세트 만들기에 대한 자세한 내용은 143페이지의 "새 스킨 만들기"을 참조하십시오.

skin 속성을 설정하거나 제작하는 동안 또는 런타임에 ActionScript에서 skin 매개 변수를 설정하여 지정하는 스킨의 경우 스킨 선택과 상관없이 색상 및 알파(투명도) 값을 지정할 수 있습니다. 제작하는 동안 색상 및 알파 값을 지정하려면 다음과 같이 [스킨 선택] 대화 상자에서 색상 선택기를 엽니다.



스킨 선택 대화 상자에서 색상 선택기

색상을 선택하려면 패널에서 색상 견본을 클릭하거나 텍스트 상자에 숫자 값을 입력하고, 알파 값을 선택하려면 슬라이더를 사용하거나 [알파] 텍스트 상자에 백분율을 입력합니다.

런타임에 색상 값과 알파 값을 지정하려면 skinBackgroundColor 및 skinBackgroundAlpha 속성을 설정합니다. skinBackgroundColor 속성은 0xRRGGBB(빨강, 녹색, 파랑) 값으로 설정하고, skinBackgroundAlpha 속성은 0.0에서 1.0 사이의 숫자로 설정합니다. 다음 예제에서는 skinBackgroundColor를 0xFF0000(빨강)으로, skinBackgroundAlpha를 0.5로 설정합니다.

```
my_FLVPlybk.skinBackgroundColor = 0xFF0000;
my_FLVPlybk.skinBackgroundAlpha = .5;
```

사용자가 마지막으로 선택한 값이 기본값입니다.

FLV Playback Custom UI 구성 요소를 사용하여 FLVPlayback 구성 요소를 스키닝하려면 팝업 메뉴에서 [없음]을 선택합니다.

개별적인 FLV Playback Custom UI 구성 요소 스키닝

FLV Playback Custom UI 구성 요소를 사용하면 FLA 파일 내에서 FLVPlayback 컨트롤의 모양을 사용자 정의할 수 있으며 웹 페이지를 미리 볼 때 결과를 확인할 수 있습니다. 다만 이 구성 요소는 크기 조절이 가능하도록 설계되지 않았습니다. 그러므로 동영상 클립과 내용을 특정 크기로 맞춰 편집해야 합니다. 따라서 지정한 크기로 스테이지에 배치한 FLVPlayback 구성 요소에 대해 scaleMode를 exactFit로 설정하는 것이 좋습니다.

먼저 [구성 요소] 패널에서 원하는 FLV Playback Custom UI 구성 요소를 드래그하여 스테이지의 원하는 위치에 배치하고 인스턴스 이름을 지정합니다.

이러한 구성 요소는 ActionScript 없이 작동할 수 있습니다. 이들을 FLVPlayback 구성 요소와 같은 타임라인과 프레임에 배치하려는데 구성 요소에 스킨이 설정되어 있지 않은 경우 FLVPlayback 구성 요소가 자동으로 이 구성 요소에 연결됩니다. 스테이지에 FLVPlayback 구성 요소가 여러 개 있거나 사용자 정의 컨트롤과 FLVPlayback 인스턴스가 같은 타임라인에 없는 경우 액션이 필요합니다.

스테이지에 구성 요소가 배치되고 나면 다른 심볼과 마찬가지 방법으로 구성 요소를 편집합니다. 구성 요소를 열면 각 구성 요소가 다른 구성 요소와 조금씩 다르게 설정되어 있는 것을 볼 수 있습니다.

버튼 구성 요소

버튼 구성 요소는 유사한 구조를 갖고 있습니다. 버튼 구성 요소로는 BackButton, ForwardButton, MuteButton, PauseButton, PlayButton, PlayPauseButton 및 StopButton이 있습니다. 이러한 버튼 구성 요소는 대부분 placeholder_mc라는 인스턴스 이름으로 프레임 1에 배치되는 단일 동영상 클립입니다. 이러한 동영상 클립은 해당 버튼의 보통 상태를 나타내는 인스턴스이지만 꼭 그런 것은 아닙니다. 프레임 2에는 각각의 표시 상태(보통, 오버, 다운, 비활성)에 대한 네 가지 스테이지 동영상 클립이 있습니다. 런타임에서 구성 요소가 프레임 2로 실제 이동하는 것은 아니며 동영상 클립을 보다 편리하게 편집하고 [심볼 속성] 대화 상자에서 [첫 프레임으로 내보내기] 체크 상자를 선택하지 않고도 SWF 파일에 동영상 클립이 강제로 로드되도록 프레임 2에 이러한 동영상 클립이 배치됩니다. 그러나 [ActionScript에 내보내기] 옵션은 여전히 선택해야 합니다.

버튼을 스키닝하려면 각각의 동영상 클립을 편집하면 됩니다. 크기뿐만 아니라 모양도 변경할 수 있습니다.

일부 ActionScript는 프레임 1에 나타나며 이 스크립트는 변경할 필요가 없습니다. 이러한 스크립트는 프레임 1에서 재생 헤드를 중단하고 상태에 따라 사용할 동영상 클립을 지정하는 역할을 합니다.

PlayPauseButton, MuteButton, FullScreenButton 및 CaptionButton 버튼

PlayPauseButton, MuteButton, FullScreenButton 및 CaptionButton 버튼은 다른 버튼과 다르게 설정됩니다. 이 버튼은 두 개의 레이어를 포함하는 하나의 프레임으로 구성되고 스크립트가 없습니다. 이 프레임에는 두 개의 버튼이 서로 겹쳐서 배치됩니다. 즉, PlayPauseButton의 경우 재생 및 일시 정지 버튼, MuteButton의 경우 음소거 켜기 및 음소거 끄기 버튼, FullScreenButton의 경우 전체 화면 설정 및 전체 화면 해제 버튼, CaptionButton의 경우 캡션 설정 및 캡션 해제 버튼이 배치됩니다. 이러한 버튼을 스키닝하려면 139페이지의 "개별적인 FLV Playback Custom UI 구성 요소 스키닝"에서 설명하는 방법에 따라 두 개의 내부 버튼을 각각 스키닝하면 됩니다. 다른 작업은 필요하지 않습니다.

CaptionButton은 FLVPlaybackCaptioning 구성 요소용이므로 FLVPlayback 구성 요소가 아닌 이 구성 요소에 연결해야 합니다.

BackButton 및 ForwardButton 버튼

BackButton 및 ForwardButton 버튼 또한 다른 버튼과 다르게 설정됩니다. 프레임 2에는 하나 또는 양쪽의 버튼 주위에 프레임으로 사용할 수 있는 추가적인 동영상 클립이 배치되어 있습니다. 이 동영상 클립은 꼭 필요한 것이 아니고 특별한 기능도 없으며 편의상 제공하는 것일 뿐입니다. 이 버튼을 사용하려면 [라이브러리] 패널에서 스테이지로 드래그하여 원하는 위치에 배치합니다. 이 버튼이 필요하지 않으면 사용하지 않거나 [라이브러리] 패널에서 삭제하면 됩니다.

대부분의 버튼이 공용 동영상 클립 세트를 기반으로 제공되기 때문에 모든 버튼의 모양을 한번에 변경할 수 있습니다. 이 기능을 사용하거나 공용 클립을 교체하여 모든 버튼을 다르게 보이도록 할 수 있습니다.

BufferingBar 구성 요소

버퍼링 막대 구성 요소는 간단하게 구성 요소가 버퍼링 상태로 진입할 때 표시되는 애니메이션으로 구성되어 있으며 이 구성 요소를 구성하는 데 특별한 ActionScript는 필요하지 않습니다. 기본적으로 이 구성 요소는 "이발소 간판" 효과를 주는 사각형 마스크가 입혀져 왼쪽에서 오른쪽으로 이동하는 줄무늬 막대 형태입니다. 이 구성에 특별한 부분은 없습니다.

스킨 SWF 파일에 있는 버퍼링 막대는 런타임에 크기 조절이 필요하기 때문에 9-슬라이스 크기 조절을 사용하지만 BufferingBar FLV Custom UI 구성 요소는 중첩된 동영상 클립을 갖고 있기 때문에 9-슬라이스 크기 조절을 사용하지 않으며 사용할 수도 없습니다. BufferingBar를 더 넓게 또는 길게 만들려면 전체 크기를 조절하는 것이 아니라 내용을 변경해야 합니다.

SeekBar 및 VolumeBar 구성 요소

SeekBar 및 VolumeBar 구성 요소는 유사해 보이지만 서로 다른 기능을 갖고 있습니다. 두 구성 요소 모두 핸들이 있으며 동일한 핸들 추적 메커니즘을 사용하고 진행률 및 채움률을 추적하는 중첩 클립을 지원합니다.

FLVPlayback 구성 요소의 ActionScript 코드에서는 SeekBar 또는 VolumeBar 구성 요소의 등록 포인트(원점이라고도 함)가 내용의 왼쪽 위 모서리에 있는 것으로 간주하는 경우가 많기 때문에 이 규칙을 따르는 것이 중요합니다. 그렇지 않으면 핸들, 진행률 및 채움률 동영상 클립을 사용하는 데 문제가 발생할 수 있습니다.

스킨 SWF 파일에 있는 검색 막대는 런타임에 크기 조절이 필요하기 때문에 9-슬라이스 크기 조절을 사용하지만 SeekBar FLV Custom UI 구성 요소는 중첩된 동영상 클립을 갖고 있기 때문에 9-슬라이스 크기 조절을 사용하지 않으며 사용할 수도 없습니다. SeekBar를 더 넓게 또는 길게 만들려면 전체 크기를 조절하는 것이 아니라 내용을 변경해야 합니다.

핸들

핸들 동영상 클립의 인스턴스는 프레임 2에 배치됩니다. BackButton 및 ForwardButton 구성 요소와 마찬가지로 구성 요소가 프레임 2로 실제 이동하는 것은 아니며 이러한 동영상 클립은 편집을 보다 간편하게 하고 [심볼 속성] 대화 상자에서 [첫 프레임으로 내보내기] 체크 상자를 선택하지 않고도 SWF 파일에 강제로 로드되도록 하기 위해 프레임 2에 배치해 놓은 것입니다. 그러나 [ActionScript에 내보내기] 옵션은 여전히 선택해야 합니다.

핸들 동영상 클립의 배경에는 알파 값이 0으로 설정된 사각형이 있습니다. 이 사각형은 핸들의 히트 영역 크기를 늘려서 버튼의 히트 상태와 유사하게 모양을 변경하지 않고도 쉽게 핸들을 잡을 수 있도록 합니다. 핸들은 런타임에 동적으로 만들어지기 때문에 버튼이 아니라 동영상 클립이어야 합니다. 알파 값이 0으로 설정된 이 사각형은 다른 목적으로 사용되는 것이 아니며 핸들 내부를 사용자가 원하는 다른 이미지로 교체할 수 있습니다. 등록 포인트를 핸들 동영상 클립의 수평 중앙에 유지하는 것이 가장 좋습니다.

다음 ActionScript 코드는 SeekBar 구성 요소의 프레임 1에 삽입되어 핸들을 관리합니다.

```
stop();
handleLinkageID = "SeekBarHandle";
handleLeftMargin = 2;
handleRightMargin = 2;
handleY = 11;
```

프레임 2에 내용이 있으므로 stop() 함수 호출이 필요합니다.

두 번째 행은 핸들로 사용할 심볼을 지정합니다. 프레임 2에 있는 핸들 동영상 클립 인스턴스를 간단히 편집하여 사용할 경우 이 부분을 변경할 필요가 없습니다. 런타임에 FLVPlayback 구성 요소가 스테이지의 지정한 동영상 클립의 인스턴스를 Bar 구성 요소 인스턴스의 형제 인스턴스로 만듭니다. 즉, 동일한 부모 동영상 클립을 갖게 됩니다. 그러므로 막대가 루트 레벨에 있는 경우 핸들 또한 루트 레벨에 있어야 합니다.

handleLeftMargin 변수는 핸들의 최초 위치(0%)를 결정하고, handleRightMargin 변수는 핸들의 마지막 위치(100%)를 결정합니다. 이 숫자를 통해 막대 컨트롤의 왼쪽 끝과 오른쪽 끝의 오프셋을 구하여 양수이면 막대 내부 구역을 표시하고 음수이면 막대 외부 구역을 표시합니다. 이 오프셋은 핸들의 등록 포인트를 기준으로 이동할 수 있는 방향을 지정합니다. 등록 포인트를 핸들의 중앙으로 지정한 경우 핸들의 왼쪽 끝과 오른쪽 끝이 막대의 영역을 지나치게 됩니다. 검색 막대 동영상 클립에서는 등록 포인트가 내용의 왼쪽 위 모서리에 있어야 정상적으로 작동됩니다.

handleY 변수는 핸들의 y 위치를 막대 인스턴스에 상대적인 위치로 결정합니다. 이 위치 값은 각 동영상 클립의 등록 포인트를 기준으로 합니다. 샘플 핸들에서 등록 포인트는 보이지 않는 히트 상태 사각형과는 관계없이 시각적으로 보이는 부분에 상대적인 위치에 배치할 수 있도록 삼각 모양의 끝 부분에 있습니다. 또한 막대 동영상 클립은 등록 포인트가 내용의 왼쪽 위 모서리에 있어야만 정상적으로 작동합니다.

예를 들어, 이러한 제한 조건에서 막대 컨트롤이 (100, 100)으로 설정되고 폭이 100픽셀일 경우 핸들의 가로 이동 범위는 102에서 198이며 세로로는 111에 위치합니다. handleLeftMargin 및 handleRightMargin을 -2로 변경하고 handleY를 -11로 변경하면 핸들의 가로 이동 범위는 98에서 202로 변경되고 세로로는 89에 위치하게 됩니다.

진행률 및 채움률 동영상 클립

SeekBar 구성 요소에는 **진행률** 동영상 클립이 있으며 VolumeBar에는 **채움률** 동영상 클립이 있지만 실제 이 구성 요소를 사용할 때는 진행률 및 채움률 동영상 클립 중 하나를 사용하거나 전혀 사용하지 않거나 둘 모두 사용할 수 있습니다. 이 두 가지 동영상 클립은 구조적으로 동일하며 유사하게 동작하지만 서로 다른 값을 추적합니다. 진행률 동영상 클립은 FLV 파일 다운로드 시 채워지는 동영상 클립(FMS에서 스트리밍하는 경우에는 항상 채워져 있는 상태이므로 HTTP 다운로드에서만 유용한 요소)이고 채움률 동영상 클립은 핸들이 왼쪽에서 오른쪽으로 이동하면서 채워지는 동영상 클립입니다.

FLVPlayback 구성 요소에서는 특정 인스턴스 이름으로 이 동영상 클립 인스턴스를 찾기 때문에 진행률 동영상 클립 인스턴스가 막대 동영상 클립을 부모로 갖고 있어야 하며 인스턴스 이름이 progress_mc여야 합니다. 채움률 동영상 클립 인스턴스의 이름은 fullness_mc여야 합니다.

진행률 및 채움률 동영상 클립 내부에 fill_mc 동영상 클립 인스턴스를 중첩시키거나 중첩시키지 않을 수 있습니다. VolumeBar fullness_mc 동영상 클립은 fill_mc 동영상 클립을 **사용하는** 방법을 보여 주며, SeekBar progress_mc 동영상 클립은 fill_mc 동영상 클립을 **사용하지 않는** 방법을 보여 줍니다.

크기를 조절하면 모양이 왜곡되는 채움을 사용하려면 내부에 중첩된 fill_mc 동영상 클립을 사용하는 방법이 유용합니다.

VolumeBar fullness_mc 동영상 클립에서는 중첩된 fill_mc 동영상 클립 인스턴스가 마스크 처리됩니다. 동영상 클립을 만들 때 마스크를 적용할 수 있습니다. 그렇지 않으면 런타임에 마스크가 동적으로 만들어집니다. 동영상 클립에 마스크를 적용할 경우 해당 인스턴스 이름을 **mask_mc**로 지정하고 100%에 도달했을 때 fill_mc에서 나타나도록 설정합니다. fill_mc에 마스크를 적용하지 않는 경우 동적으로 만들어진 마스크는 사각형 모양으로 100%에서 fill_mc와 동일한 크기로 만들어집니다.

fill_mc.slideReveal의 값이 true인지 false인지에 따라 마스크가 적용된 fill_mc 동영상 클립이 다음 두 가지 방법 중 하나로 표시됩니다.

fill_mc.slideReveal이 true인 경우 fill_mc가 왼쪽에서 오른쪽으로 이동하면서 마스크를 통해 표시됩니다. 0%에서는 왼쪽 끝에 있으므로 아무것도 마스크를 통해 표시되지 않습니다. 백분율이 증가함에 따라 100%에 도달할 때까지 계속 오른쪽으로 이동하며, 스테이지에서 만들어진 위치로 되돌아갑니다.

fill_mc.slideReveal이 false 또는 undefined인 경우(기본 비헤이비어) 마스크가 왼쪽에서 오른쪽으로 크기 조절되면서 fill_mc가 차츰 표시됩니다. 0%에서는 마스크가 가로로 05만큼 크기 조절되며 백분율이 증가함에 따라 scaleX가 증가하고 100%에 도달하면 fill_mc 전체가 표시됩니다. mask_mc가 만들어질 때 이미 크기가 조절되었을 수 있으므로 항상 scaleX = 100을 의미하는 것은 아닙니다.

fill_mc를 사용하지 않는 방법이 fill_mc를 사용하는 방법보다 간편하지만 가로 채움 모양이 왜곡됩니다. 왜곡 현상을 원하지 않는다면 fill_mc를 사용해야 합니다. SeekBar progress_mc에서는 이 방법을 보여 줍니다.

진행률 또는 채움률 동영상 클립은 백분율에 따라 가로로 크기 조절됩니다. 0%에서는 인스턴스의 scaleX가 0으로 설정되며 인스턴스가 화면에 보이지 않습니다. 백분율이 증가함에 따라 scaleX가 조절되며 100%에 도달하면 동영상 클립이 처음 스테이지에 만들어졌을 때와 동일한 크기가 됩니다. 앞에서와 마찬가지로 동영상 클립 인스턴스가 만들어졌을 때 이미 크기가 조절되었을 수 있으므로 항상 scaleX = 100을 의미하는 것은 아닙니다.

FLV Playback Custom UI 구성 요소 연결

Custom UI 구성 요소를 FLVPlayback 구성 요소와 같은 타임라인과 프레임에 배치하려는데 skin 속성이 설정되어 있지 않은 경우 ActionScript 없이도 FLVPlayback이 자동으로 Custom UI 구성 요소에 연결됩니다.

스테이지에 FLVPlayback 구성 요소가 여러 개 있거나 사용자 정의 컨트롤과 FLVPlayback이 같은 타임라인에 없는 경우 ActionScript 코드를 작성하여 Custom UI 구성 요소를 FLVPlayback 구성 요소의 인스턴스에 연결해야 합니다. 먼저 FLVPlayback 인스턴스에 이름을 지정한 다음 ActionScript를 사용하여 FLV Playback Custom UI 구성 요소 인스턴스를 해당되는 FLVPlayback 속성에 지정해야 합니다. 다음 예제에서 FLVPlayback 인스턴스는 my_FLVPlybk이고 마침표(.) 뒤에 쓰여진 것이 FLVPlayback 속성 이름이며 등호 기호(=) 오른쪽에 있는 것이 FLV Playback Custom UI 컨트롤입니다.

```
//FLVPlayback instance = my_FLVPlybk
my_FLVPlybk.playButton = playbtn; // set playButton prop. to playbtn, etc.
my_FLVPlybk.pauseButton = pausebtn;
my_FLVPlybk.playPauseButton = playpausebtn;
my_FLVPlybk.stopButton = stopbtn;
my_FLVPlybk.muteButton = mutebtn;
my_FLVPlybk.backButton = backbtn;
my_FLVPlybk.forwardButton = forbtn;
my_FLVPlybk.volumeBar = volbar;
my_FLVPlybk.seekBar = seekbar;
my_FLVPlybk.bufferingBar = bufbar;
```

다음 단계에서는 사용자 정의 StopButton, PlayPauseButton, MuteButton 및 SeekBar 컨트롤을 만듭니다.

1. FLVPlayback 구성 요소를 스테이지로 드래그하고 인스턴스 이름으로 **my_FLVPlybk**를 지정합니다.
2. 구성 요소 관리자에서 source 매개 변수를 **http://www.helpexamples.com/flash/video/cuepoints.flv**로 설정합니다.
3. Skin 매개 변수를 [없음]으로 설정합니다.
4. StopButton, PlayPauseButton 및 MuteButton을 각각 하나씩 스테이지로 드래그하고 FLVPlayback 인스턴스 위에 수직으로 왼쪽에 쌓습니다. 속성 관리자에서 각 버튼에 **my_stopbttn**, **my_plypausbttn** 및 **my_mutebttn**과 같은 인스턴스 이름을 지정합니다.
5. [라이브러리] 패널에서 FLVPlayback Skins 폴더를 열고 이 폴더 아래에 있는 SquareButton 폴더를 엽니다.
6. SquareBgDown 동영상 클립을 선택하고 두 번 클릭하여 스테이지에서 엽니다.
7. 마우스 오른쪽 버튼을 클릭(Windows)하거나 Control 키를 누른 상태에서 클릭(Macintosh)하고 메뉴에서 [전체 선택]을 선택한 후 심볼을 삭제합니다.

8 [타원형 도구]를 선택하고 같은 위치에 타원을 그린 다음 채움 색상을 파랑(**#0033FF**)으로 설정합니다.

9 속성 관리자에서 폭(W:)은 **40**으로 설정하고 높이(H:)는 **20**으로 설정합니다. X 좌표(X:)는 **0.0**으로 설정하고 Y 좌표(Y:)는 **0.0**으로 설정합니다.

10 SquareBgNormal에 대해서도 6~8단계를 반복하되 채움 색상은 노랑(**#FFFF00**)으로 변경합니다.

11 SquareBgOver에 대해서도 6~8단계를 반복하되 채움 색상은 녹색(**#006600**)으로 변경합니다.

12 버튼 내부에 있는 다양한 심볼 아이콘(PauseIcon, PlayIcon, MuteOnIcon, MuteOffIcon, StopIcon)에 대한 동영상 클립을 편집합니다. 이러한 동영상 클립은 [라이브러리] 패널에서 FLV Playback Skins/**Label** Button/Assets에서 찾을 수 있습니다. 여기서 **Label**은 Play, Pause 등과 같은 버튼의 이름입니다. 각 항목에 대해 다음 단계를 수행합니다.

a [전체 선택] 옵션을 선택합니다.

b 색상을 빨강(**#FF0000**)으로 변경합니다.

c 300%로 크기 조절합니다.

d 내용의 X: 위치를 **7.0**으로 변경하여 모든 버튼 상태에서 아이콘의 가로 위치를 수정합니다.

참고: 위치를 이와 같은 방법으로 변경하면 각각의 버튼 상태를 열고 아이콘 동영상 클립 인스턴스를 이동하는 번거로움을 피할 수 있습니다.

13 타임라인 위의 파란 색 [뒤로] 화살표를 클릭하여 장면 1, 프레임 1로 돌아갑니다.

14 SeekBar 구성 요소를 스테이지로 드래그하고 FLVPlayback 인스턴스의 오른쪽 아래 모서리에 배치합니다.

15 [라이브러리] 패널에서 SeekBar를 두 번 클릭하여 스테이지에서 엽니다.

16 400%로 크기 조절합니다.

17 외곽선을 선택하고 색상을 빨강(**#FF0000**)으로 설정합니다.

18 FLVPlayback Skins/Seek Bar 폴더에서 SeekBarProgress를 두 번 클릭하고 색상을 노랑(**#FFFF00**)으로 설정합니다.

19 FLVPlayback Skins/Seek Bar 폴더에서 SeekBarHandle을 두 번 클릭하고 색상을 빨강(**#FF0000**)으로 설정합니다.

20 타임라인 위의 파란 색 [뒤로] 화살표를 클릭하여 장면 1, 프레임 1로 돌아갑니다.

21 스테이지에서 SeekBar 인스턴스를 선택하고 인스턴스 이름으로 **my_seekbar**를 지정합니다.

22 타임라인의 프레임 1에서 [액션] 패널을 열고 다음 예제와 같이 비디오 클래스에 대한 import 명령문을 추가하고 버튼 및 검색 막대 이름을 해당하는 FLVPlayback 속성에 지정합니다.

```
import fl.video.*;
my_FLVPlybk.stopButton = my_stopbttn;
my_FLVPlybk.playPauseButton = my_plypausbttn;
my_FLVPlybk.muteButton = my_mutebttn;
my_FLVPlybk.seekBar = my_seekbar;
```

23 Ctrl+Enter를 눌러 동영상을 테스트합니다.

새 스킨 만들기

스킨 SWF 파일을 만드는 가장 좋은 방법은 Flash에서 제공하는 스킨 파일 중 하나를 복사해서 편집하는 것입니다. 이러한 스킨에 대한 FLA 파일은 Flash 응용 프로그램 폴더의 Configuration/FLVPlayback Skins/FLA/ActionScript 3.0/ 폴더에서 찾을 수 있습니다. 사용자가 제작한 스킨 SWF 파일을 [스킨 선택] 대화 상자에서 선택할 수 있는 옵션으로 나타나게 하려면 Flash 응용 프로그램 폴더의 Configuration/FLVPlayback Skins/ActionScript 3.0 폴더 또는 사용자의 로컬 Configuration/FLVPlayback Skins/ActionScript 3.0 폴더에 해당 스킨 SWF 파일을 배치합니다.

스킨 색상은 스킨 선택과 관계없이 설정할 수 있으므로 FLA 파일을 편집하여 색상을 수정할 필요가 없습니다. 특정 색상의 스킨을 만드는 경우 [스킨 선택] 대화 상자에서 해당 스킨을 편집하지 못하게 하려면 스킨 FLA ActionScript 코드에서 this.border_mc.colorMe = false;를 설정합니다. 스킨 색상 설정에 대한 자세한 내용은 138페이지의 “미리 제작된 스킨 선택”을 참조하십시오.

설치된 Flash 스킨 FLA 파일을 보면 스테이지에 배치된 요소 중 일부는 필요 없는 것처럼 보이지만 이러한 요소 중 많은 부분이 안내 레이어에 배치되는 것입니다. 스케일 9로 실시간 미리 보기를 사용하면 SWF 파일에 실제로 표시되는 내용을 빠르게 확인할 수 있습니다.

다음 단원에서는 SeekBar, BufferingBar 및 VolumeBar 동영상 클립에 대한 보다 복잡한 사용자 정의 및 변경 방법에 대해 설명합니다.

스킨 레이아웃 사용

Flash 스킨 FLA 파일을 열면 스킨의 동영상 클립이 기본 타임라인에 배치되어 있습니다. 동일한 프레임에 있는 이러한 클립과 ActionScript 코드는 런타임에 컨트롤을 배치하는 방법을 정의합니다.

레이아웃 레이어는 런타임에 표시되는 스킨의 모양과 많이 유사하지만 이 레이어의 내용은 런타임에 보이지 않습니다. 이 클립은 컨트롤을 배치하는 위치를 계산하기 위한 목적으로만 사용됩니다. 스테이지의 다른 컨트롤은 런타임에 사용됩니다.

레이아웃 레이어 내부에는 video_mc라는 FLVPlayback 구성 요소에 대한 자리 표시자가 있습니다. 다른 모든 컨트롤은 video_mc에 상대적인 위치로 배치됩니다. Flash FLA 파일을 기초로 컨트롤 크기를 변경하는 경우 이러한 자리 표시자 클립을 이동하여 레이아웃을 수정할 수 있습니다.

각각의 자리 표시자 클립은 특정 인스턴스 이름을 갖고 있습니다. 자리 표시자 클립의 이름은 playpause_mc, play_mc, pause_mc, stop_mc, captionToggle_mc, fullScreenToggle_mc, back_mc, bufferingBar_mc, bufferingBarFill_mc, seekBar_mc, seekBarHandle_mc, seekBarProgress_mc, volumeMute_mc, volumeBar_mc 및 volumeBarHandle_mc입니다. 스킨 색상을 선택할 때 색상이 변경되는 부분은 border_mc입니다.

컨트롤에 어떤 클립이 사용되는지는 중요하지 않습니다. 일반적으로 버튼에는 보통 상태 클립이 사용됩니다. 다른 컨트롤에 대해서는 해당 컨트롤의 클립이 사용되지만 이러한 설정은 단순히 편의를 위한 것일 뿐입니다. 중요한 것은 자리 표시자의 **x**(가로) 및 **y**(세로) 위치와 높이 및 폭 값입니다.

표준 컨트롤 외에도 클립을 추가로 원하는 만큼 여러 개 배치할 수도 있습니다. 이러한 클립에 대한 유일한 요구 사항은 [링크] 대화 상자에서 라이브러리 심볼의 [ActionScript에 내보내기] 체크 상자가 선택되어 있어야 한다는 것입니다. 레이아웃 레이어의 사용자 정의 클립은 위에 나열된 예약 인스턴스 이름이 아닌 다른 인스턴스 이름을 가질 수 있습니다. 인스턴스 이름은 클립의 ActionScript를 설정하여 레이아웃을 결정할 때만 필요합니다.

border_mc 클립은 특별한 클립입니다. FlvPlayback.skinAutoHide 속성을 true로 설정하면 마우스가 border_mc 클립 위에 있을 때 스킨이 표시됩니다. 이러한 동작은 비디오 플레이어의 경계 밖에 나타나는 스킨에 중요합니다. skinAutoHide 속성에 대한 자세한 내용은 147페이지의 “스킨 비헤이비어 수정”을 참조하십시오.

Flash FLA 파일에서 border_mc는 크롬과 Forward 및 Back 버튼의 테두리에 사용됩니다.

border_mc 클립은 또한 skinBackgroundAlpha 및 skinBackgroundColor 속성에 의해 알파 및 색상이 변경된 스킨의 일부분이기도 합니다. 사용자 정의 가능한 색상과 알파를 허용하려면 스킨 FLA 파일의 ActionScript에 다음이 포함되어야 합니다.

```
border_mc.colorMe = true;
```

ActionScript 및 스킨 레이아웃

다음 ActionScript 코드는 일반적으로 모든 컨트롤에 적용됩니다. 일부 컨트롤은 추가적인 비헤이비어를 정의하는 특정 ActionScript를 갖고 있습니다. 이 부분에 대해서는 해당 컨트롤을 설명하는 단원에서 설명합니다.

초기 ActionScript는 각 구성 요소의 개별 상태에 대한 클래스 이름을 지정하는 큰 섹션입니다. 이러한 모든 클래스 이름은 SkinOverAll.fla 파일에서 찾을 수 있습니다. 예를 들어, 일시 정지 및 재생 버튼에 대한 코드는 다음과 같습니다.

```
this.pauseButtonDisabledState = "fl.video.skin.PauseButtonDisabled";
this.pauseButtonDownState = "fl.video.skin.PauseButtonDown";
this.pauseButtonNormalState = "fl.video.skin.PauseButtonNormal";
this.pauseButtonOverState = "fl.video.skin.PauseButtonOver";
this.playButtonDisabledState = "fl.video.skin.PlayButtonDisabled";
this.playButtonDownState = "fl.video.skin.PlayButtonDown";
this.playButtonNormalState = "fl.video.skin.PlayButtonNormal";
this.playButtonOverState = "fl.video.skin.PlayButtonOver";
```

클래스 이름은 실제 외부 클래스 파일을 갖지 않으므로 라이브러리에 있는 모든 동영상 클립에 대한 클래스 이름은 [링크] 대화 상자에 지정되어 있습니다.

ActionScript 2.0 구성 요소의 경우 런타임에 실제로 사용되는 스테이지 동영상 클립이 있습니다. ActionScript 3.0 구성 요소의 경우 이러한 동영상 클립이 계속 FLA 파일에 있지만 이는 보다 편리하게 편집하기 위한 것입니다. 이제 이러한 동영상 클립은 모두 안내 레이어이므로 내보낼 수 없습니다. 라이브러리에 있는 모든 스킨 에셋은 첫 번째 프레임에서 내보내지도록 설정되며 다음과 같은 코드를 사용하여 동적으로 만들어집니다.

```
new fl.video.skin.PauseButtonDisabled();
```

다음 섹션은 스킨의 최소 폭과 높이를 정의하는 ActionScript 코드입니다. 이 값은 [스킨 선택] 대화 상자에 표시되며 해당 스킨이 최소 크기 이하로 조절되지 않도록 런타임에 사용됩니다. 최소 크기를 지정하지 않으려면 이 값을 undefined로 두거나 0 또는 0보다 작은 값으로 지정합니다.

```
// minimum width and height of video recommended to use this skin,
// leave as undefined or <= 0 if there is no minimum
this.minWidth = 270;
this.minHeight = 60;
```

각 자리 표시자에는 다음과 같은 속성이 적용될 수 있습니다.

속성	설명
anchorLeft	부울. FLVPlayback 인스턴스의 왼쪽을 기준으로 상대적인 위치에 컨트롤을 배치합니다. anchorRight가 명시적으로 true로 설정된 경우에는 false로 기본 설정되지만, 그렇지 않은 경우에는 true로 기본 설정됩니다.
anchorRight	부울. FLVPlayback 인스턴스의 오른쪽을 기준으로 상대적인 위치에 컨트롤을 배치합니다. 기본값은 false입니다.
anchorBottom	부울. FLVPlayback 인스턴스의 아래쪽을 기준으로 상대적인 위치에 컨트롤을 배치합니다. anchorTop이 명시적으로 true로 설정된 경우에는 false로 기본 설정되지만 그렇지 않은 경우 true로 기본 설정됩니다.
anchorTop	부울. FLVPlayback 인스턴스의 위쪽을 기준으로 상대적인 위치에 컨트롤을 배치합니다. 기본값은 false입니다.

anchorLeft 및 anchorRight 속성이 모두 true인 경우 이 컨트롤은 런타임에 가로로 크기 조절됩니다. anchorTop 및 anchorBottom 속성이 모두 true인 경우 이 컨트롤은 런타임에 세로로 크기 조절됩니다.

이 속성의 효과를 확인하려면 이 속성이 Flash 스킨에서 어떻게 사용되는지 살펴보십시오. BufferingBar 및 SeekBar 컨트롤만이 크기 조절되는 컨트롤이며 차례로 포개져서 배치되고 anchorLeft 및 anchorRight 속성이 모두 true로 설정됩니다. BufferingBar 및 SeekBar 왼쪽에 있는 모든 컨트롤은 anchorLeft 속성이 true로 설정되고 오른쪽에 있는 컨트롤은 anchorRight 속성이 true로 설정됩니다. 모든 컨트롤의 anchorBottom 속성은 true로 설정됩니다.

스킨에서 컨트롤을 아래쪽이 아니라 위쪽에 배치하도록 레이아웃 레이어의 동영상 클립을 편집할 수 있습니다. 이렇게 하려면 컨트롤을 video_mc를 기준으로 위쪽으로 이동하고 모든 컨트롤에 대해 anchorTop을 true로 설정합니다.

버퍼링 막대

버퍼링 막대에는 bufferingBar_mc와 bufferingBarFill_mc라는 두 가지 동영상 클립이 있습니다. 두 클립의 상대적인 위치는 계속 유지되기 때문에 스테이지에서 두 클립의 위치 설정은 중요합니다. 구성 요소에서 bufferingBarFill_mc는 크기 조절하지 않고 bufferingBar_mc를 크기 조절하기 때문에 버퍼링 막대는 두 개의 개별적인 클립을 사용합니다.

bufferingBar_mc 클립은 9-슬라이스 크기 조절이 적용되기 때문에 크기가 조절될 때 테두리가 왜곡되지 않습니다. bufferingBarFill_mc 클립은 상당히 넓기 때문에 크기 조절이 필요 없을 만큼 충분히 넓습니다. 런타임에 자동으로 마스크가 적용되어 확장된 bufferingBar_mc 위에 해당 부분만 표시됩니다. 기본적으로 마스크의 정확한 크기는 bufferingBar_mc와 bufferingBarFill_mc의 x(가로) 위치 사이의 차이 값을 기준으로 bufferingBar_mc 내에서 왼쪽 및 오른쪽 여백과 동일하도록 유지됩니다. ActionScript 코드를 사용하여 이 위치를 사용자 정의할 수 있습니다.

버퍼링 막대에서 크기 조절이 필요하지 않거나 9-슬라이스 크기 조절을 사용하지 않는 경우 FLV Playback Custom UI BufferingBar 구성 요소처럼 설정할 수 있습니다. 자세한 내용은 140페이지의 “BufferingBar 구성 요소”을 참조하십시오.

버퍼링 막대에는 다음과 같은 추가 속성이 있습니다.

속성	설명
fill_mc:MovieClip	버퍼링 막대 채움 인스턴스 이름을 지정합니다. 기본값은 bufferingBarFill_mc입니다.

검색 막대 및 볼륨 막대

검색 막대에도 seekBar_mc와 seekBarProgess_mc라는 두 가지 동영상 클립이 있습니다. 두 클립의 상대적인 위치는 계속 유지되기 때문에 레이아웃 레이어에서 두 클립의 위치 설정은 중요합니다. 두 가지 클립 모두 크기 조절이 가능하지만 seekBar_mc는 9-슬라이스 크기 조절을 사용하며 9-슬라이스 크기 조절은 중첩된 동영상 클립에서 정상적으로 작동되지 않기 때문에 seekBarProgress_mc는 seekBar_mc 내부에 중첩될 수 없습니다.

seekBar_mc 클립은 9-슬라이스 크기 조절이 적용되기 때문에 크기가 조절될 때 테두리가 왜곡되지 않습니다. seekBarProgress_mc 클립도 크기 조절이 가능하지만 테두리가 왜곡됩니다. 이 클립은 채움 클립이기 때문에 9-슬라이스를 사용하지 않지만 왜곡되더라도 보기가 나쁘지 않습니다.

seekBarProgress_mc 클립은 fill_mc 없이 사용할 수 있으며 progress_mc 클립을 FLV Playback Custom UI 구성 요소에서 사용하는 것과 유사합니다. 즉, 마스크가 적용되지 않으며 가로로 크기 조절됩니다. 100%에서 seekBarProgress_mc의 정확한 크기는 seekBarProgress_mc 클립의 왼쪽과 오른쪽 여백에 의해 정의됩니다. 이 크기는 기본적으로 동일하며 seekBar_mc와 seekBarProgress_mc의 **x**(가로) 위치의 차이 값을 기준으로 정의됩니다. 검색 막대 동영상 클립에서 다음 예제와 같이 ActionScript를 사용하여 크기를 사용자 정의할 수 있습니다.

```
this.seekBar_mc.progressLeftMargin = 2;
this.seekBar_mc.progressRightMargin = 2;
this.seekBar_mc.progressY = 11;
this.seekBar_mc.fullnessLeftMargin = 2;
this.seekBar_mc.fullnessRightMargin = 2;
this.seekBar_mc.fullnessY = 11;
```

이 코드를 SeekBar 동영상 클립 타임라인에 배치하거나 다른 ActionScript 코드와 함께 기본 타임라인에 배치할 수 있습니다. 레이아웃을 수정하는 대신 코드로 사용자 정의하는 경우 채움은 스테이지에 있을 필요가 없고, 올바른 클래스 이름의 프레임 1에서 ActionScript에 내보내도록 설정된 상태로 라이브러리에 있으면 됩니다.

FLV Playback Custom UI SeekBar 구성 요소처럼 검색 막대에 채움률 동영상 클립을 만들 수 있습니다. 검색 막대에서 크기 조절이 필요하지 않거나 크기는 조절하지만 9-슬라이스 크기 조절은 사용하지 않는 경우 FLV Playback Custom UI 구성 요소에 대해 사용하는 모든 메서드를 사용하여 progress_mc 또는 fullness_mc를 설정할 수 있습니다. 자세한 내용은 을 참조하십시오.

Flash 스킨의 볼륨 막대는 크기 조절되지 않기 때문에 VolumeBar FLV Playback Custom UI 구성 요소와 동일한 방식으로 구성됩니다. 자세한 내용은 140페이지의 “SeekBar 및 VolumeBar 구성 요소”을 참조하십시오. 단 핸들의 구현 방식은 다릅니다.

Seekbar 및 VolumeBar 핸들

SeekBar 및 VolumeBar 핸들은 레이아웃 레이어에서 막대 옆에 배치됩니다. 기본적으로 핸들의 왼쪽 여백, 오른쪽 여백 및 **y** 축 값은 막대 동영상 클립에 상대적인 위치로 설정됩니다. 왼쪽 여백은 핸들의 **x**(가로) 위치와 막대의 **x**(가로) 위치 사이의 차이 값으로 설정되며 오른쪽 여백은 왼쪽 여백과 동일합니다. 이 값은 SeekBar 또는 VolumeBar 동영상 클립에서 ActionScript를 사용하여 사용자 정의할 수 있습니다. 다음 예제는 FLV Playback Custom UI 구성 요소에서 사용되는 것과 동일한 ActionScript 코드입니다.

```
this.seekBar_mc.handleLeftMargin = 2;
this.seekBar_mc.handleRightMargin = 2;
this.seekBar_mc.handleY = 11;
```

이 코드를 SeekBar 동영상 클립 타임라인에 배치하거나 다른 ActionScript 코드와 함께 기본 타임라인에 배치할 수 있습니다. 레이아웃을 수정하는 대신 코드로 사용자 정의하는 경우 핸들은 스테이지에 있을 필요가 없고, 올바른 클래스 이름의 프레임 1에서 ActionScript에 내보내도록 설정된 상태로 라이브러리에 있으면 됩니다.

이러한 속성 설정 작업만 제외하면 핸들은 간단한 동영상 클립이고 FLV Playback Custom UI 구성 요소에서와 동일한 방법으로 설정하면 됩니다. 두 핸들 모두 alpha 속성이 0으로 설정된 사각형 배경을 갖고 있으며 이 사각형 배경은 히트 영역을 넓혀 주기 위한 목적뿐이므로 꼭 필요한 것은 아닙니다.

배경 및 전경 클립

동영상 클립 chrome_mc 및 forwardBackBorder_mc는 배경 클립으로 구현됩니다.

스테이지와 자리 표시자 Forward 및 Back 버튼에서 ForwardBackBorder, ForwardBorder 및 BackBorder 동영상 클립 중 안내 레이어에 배치되지 않는 것은 ForwardBackBorder뿐입니다. 이 동영상 클립은 Forward 및 Back 버튼을 실제로 사용하는 스킨에만 있습니다.

이러한 클립에 대한 유일한 요구 사항은 라이브러리의 프레임 1에서 클립을 ActionScript에 내보내야 한다는 것입니다.

스킨 비헤이비어 수정

bufferingBarHidesAndDisablesOthers 속성 및 skinAutoHide 속성을 사용하면 FLVPlayback 스킨의 비헤이비어를 사용자 정의할 수 있습니다.

bufferingBarHidesAndDisablesOthers 속성을 true로 설정하면 FLVPlayback 구성 요소가 버퍼링 상태로 바뀔 때 이 구성 요소에서 SeekBar 및 핸들을 숨기고 Play 및 Pause 버튼을 비활성화합니다. 이 방법은 느린 속도의 연결로 bufferTime 속성은 길게(예: 10) 설정하여 FLV 파일을 FMS에서 스트리밍하는 경우에 유용합니다. 속도는 느리고 버퍼링 시간은 길게 설정된 상황에서 인내심이 부족한 사용자는 Play 및 Pause 버튼을 클릭하여 검색을 시작하려고 하겠지만 이는 파일 재생을 더욱 느리게만 할 뿐입니다. 이러한 사용자의 동작을 방지하기 위해 bufferingBarHidesAndDisablesOthers 속성을 true로 설정하여 구성 요소가 버퍼링 상태에 있을 때 SeekBar 요소와 Pause 및 Play 버튼을 비활성화할 수 있습니다.

skinAutoHide 속성은 미리 제작된 스킨 SWF 파일에만 영향을 주며 FLV Playback Custom UI 구성 요소에서 만든 컨트롤에는 영향을 주지 않습니다. 이 속성이 true로 설정되어 있으면 FLVPlayback 구성 요소에서는 마우스가 표시 영역 위에 있지 않을 때 스킨을 숨깁니다. 이 속성의 기본값은 false입니다.

SMIL 파일 사용

다양한 대역폭에서 여러 스트림을 처리하기 위해 VideoPlayer 클래스에서는 SMIL의 하위 집합을 지원하는 도우미 클래스(NCManager)를 사용합니다. SMIL은 비디오 스트림의 위치, FLV 파일의 레이아웃(폭 및 높이) 및 각각의 대역폭에 해당되는 소스 FLV 파일을 식별하는 데 사용됩니다. FLV 파일의 비트율 및 지속 시간을 지정하는 데에도 사용됩니다.

source 매개 변수 또는 FLVPlayback.source 속성(ActionScript)을 사용하여 SMIL 파일의 위치를 지정할 수 있습니다. 자세한 내용은 **Adobe Flash Professional CS5**용 **ActionScript 3.0** 참조 설명서에서 및 FLVPlayback.source 속성을 참조하십시오.

다음 예제에서는 RTMP를 사용하여 FMS에서 다양한 대역폭의 FLV 파일을 스트리밍하는 SMIL 파일을 보여 줍니다.

```
<smil>
<head>
<meta base="rtmp://myserver/myapp/" />
<layout>
<root-layout width="240" height="180" />
</layout>
</head>
<body>
            <switch>
                    <ref src="myvideo_cable.flv" dur="3:00.1"/>
                    <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
                    <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1"/>
            </switch>
</body>
</smil>
```

<head> 태그에는 <meta> 및 <layout> 태그가 포함될 수 있습니다. <meta> 태그는 스트리밍 비디오(FCS의 RTMP)의 URL을 지정하는 데 사용되는 base 특성만 지원합니다.

<layout> 태그는 root-layout 요소만 지원합니다. 이 요소는 height 및 width 특성을 설정하는 데 사용되며 FLV 파일이 렌더링되는 윈도우의 크기를 결정합니다. 이 특성에는 백분율이 아닌 픽셀 값만 사용할 수 있습니다.

SMIL 파일의 본문에 FLV 소스 파일에 대한 단일 링크를 포함하거나 이전 예제와 같이 FMS에서 다양한 대역폭에 맞춰 여러 파일을 스트리밍하는 경우 <switch> 태그를 사용하여 소스 파일을 나열할 수 있습니다.

<switch> 태그 내에 사용되는 video 및 ref 태그는 같은 기능으로 둘 다 src 특성을 사용하여 FLV 파일을 지정할 수 있습니다. 또한 각 태그에 region, system-bitrate 및 dur 특성을 사용하여 지역, 필요한 최소 대역폭 및 FLV 파일의 지속 시간을 지정할 수 있습니다.

<body> 태그 내에는 <video>, <src> 또는 <switch> 태그 중 하나만 사용할 수 있습니다.

다음 예제에서는 대역폭을 사용하지 않는 단일 FLV 파일에 대한 점진적 다운로드를 보여 줍니다.

```
<smil>
        <head>
            <layout>
                <root-layout width="240" height="180" />
            </layout>
        </head>
        <body>
            <video src=""myvideo.flv" />
        </body>
</smil>
```

<smil>

지원 버전
Flash Professional 8

구문

```
<smil>
...
child tags
...
</smil>
```

특성
없음

자식 태그
<head>, <body>

부모 태그
없음

설명
SMIL 파일을 식별하는 최상위 태그입니다.

예제
다음 예제에서는 세 개의 FLV 파일을 지정하는 SMIL 파일을 보여 줍니다.

```
<smil>
<head>
<meta base="rtmp://myserver/myapp/" />
<layout>
<root-layout width="240" height="180" />
</layout>
</head>
<body>
<switch>
                <ref src="myvideo_cable.flv" dur="3:00.1"/>
                <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
                <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1"/>
            </switch>
</body>
</smil>
```

<head>

지원 버전
Flash Professional 8

구문

```
<head>
...
child tags
...
</head>
```

특성
없음

자식 태그
<meta>, <layout>

부모 태그
<smil>

설명
<meta> 및 <layout> 태그를 지원하며 소스 FLV 파일의 위치와 기본 레이아웃(높이 및 폭)을 지정합니다.

예제
다음 예제는 루트 레이아웃을 240x180픽셀로 설정합니다.

```
<head>
    <meta base="rtmp://myserver/myapp/" />
    <layout>
        <root-layout width="240" height="180" />
    </layout>
</head>
```

<meta>

지원 버전
Flash Professional 8

구문
```
<meta/>
```

특성
base

자식 태그
<layout>

부모 태그
없음

설명
소스 FLV 파일의 위치(RTMP URL)를 지정하는 base 특성을 포함합니다.

예제
다음 예제에서는 myserver의 기본 위치를 지정하는 meta 태그를 보여 줍니다.

```
<meta base="rtmp://myserver/myapp/" />
```

<layout>

지원 버전
Flash Professional 8

구문
```
<layout>
...
child tags
...
</layout>
```

특성
없음

자식 태그
<root-layout>

부모 태그
<meta>

설명
FLV 파일의 폭과 높이를 지정합니다.

예제
다음 예제에서는 레이아웃을 240픽셀 x 180픽셀로 지정합니다.

```
<layout>
    <root-layout width="240" height="180" />
</layout>
```

<root-layout>

지원 버전
Flash Professional 8

구문
```
<root-layout...attributes.../>
```

특성
width, height

자식 태그
없음

부모 태그
```
<layout>
```

설명
FLV 파일의 폭과 높이를 지정합니다.

예제
다음 예제에서는 레이아웃을 240픽셀 x 180픽셀로 지정합니다.

```
<root-layout width="240" height="180" />
```

<body>

지원 버전
Flash Professional 8

구문
```
<body>
...
child tags
...
</body>
```

특성
없음

자식 태그
```
<video>, <ref>, <switch>
```

부모 태그

```
<smil>
```

설명

소스 FLV 파일의 이름과 최소 대역폭 및 FLV 파일의 지속 시간을 지정하는 <video>, <ref> 및 <switch> 태그를 포함합니다. system-bitrate 특성은 <switch> 태그를 사용하는 경우에만 지원됩니다. <body> 태그 내에는 <switch>, <video> 또는 <ref> 태그 중 하나의 인스턴스만 사용할 수 있습니다.

예제

다음 예제에서는 세 개의 FLV 파일을 지정하는데 그 중 두 개는 video 태그를 사용하여 지정하고 다른 하나는 ref 태그를 사용하여 지정합니다.

```
<body>
    <switch>
        <ref src="myvideo_cable.flv" dur="3:00.1"/>
        <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
        <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1"/>
    </switch>
</body>
```

<video>

지원 버전

Flash Professional 8

구문

```
<video...attributes.../>
```

특성

```
src, system-bitrate, dur
```

자식 태그

없음

부모 태그

```
<body>
```

설명

<ref> 태그와 동일합니다. 소스 FLV 파일의 이름과 지속 시간을 지정하는 src 및 dur 특성을 지원합니다. dur 특성은 전체 시간 형식(00:03:00:01) 및 부분 시간 형식(03:00:01)을 지원합니다.

예제

다음 예제에서는 비디오의 소스 파일 및 지속 시간을 설정합니다.

```
<video src="myvideo_mdm.flv" dur="3:00.1"/>
```

<ref>

지원 버전
Flash Professional 8

구문
```
<ref...attributes.../>
```

특성
```
src, system-bitrate, dur
```

자식 태그
없음

부모 태그
```
<body>
```

설명
<video> 태그와 동일합니다. 소스 FLV 파일의 이름과 지속 시간을 지정하는 src 및 dur 특성을 지원합니다. dur 특성은 전체 시간 형식(00:03:00:01) 및 부분 시간 형식(03:00:01)을 지원합니다.

예제
다음 예제에서는 비디오의 소스 파일 및 지속 시간을 설정합니다.

```
<ref src="myvideo_cable.flv" dur="3:00.1"/>
```

<switch>

지원 버전
Flash Professional 8

구문
```
<switch>
...
child tags
...
<switch/>
```

특성
없음

자식 태그
```
<video>, <ref>
```

부모 태그
```
<body>
```

설명

<video> 또는 <ref> 자식 태그와 함께 사용되어 여러 대역폭 비디오 스트리밍에 대한 FLV 파일을 나열합니다. <switch> 태그는 system-bitrate 특성을 지원합니다. 이 특성은 최소 대역폭뿐만 아니라 src 및 dur 특성도 지정합니다.

예제

다음 예제에서는 세 개의 FLV 파일을 지정하는데 그 중 두 개는 video 태그를 사용하여 지정하고 다른 하나는 ref 태그를 사용하여 지정합니다.

```
<switch>
    <ref src="myvideo_cable.flv" dur="3:00.1"/>
    <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
    <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1" />
</switch>
```

7장: FLVPlayback Captioning 구성 요소 사용

FLVPlayback 구성 요소를 사용하면 다운로드한 Adobe Flash Video(FLV 또는 F4V) 파일 및 스트리밍 FLV 또는 F4V 파일을 재생하기 위해 비디오 플레이어를 Adobe Flash CS5 Professional 응용 프로그램에 포함할 수 있습니다. FLVPlayback에 대한 자세한 내용은 122페이지의 “FLVPlayback 구성 요소 사용”을 참조하십시오.

FLVPlaybackCaptioning 구성 요소를 사용하면 비디오에 대한 클로즈드 캡션 기능을 지원할 수 있습니다. 캡션 구성 요소는 W3C 표준 XML 형식의 Timed Text를 지원하며 다음과 같은 기능을 제공합니다.

포함된 이벤트 큐 포인트를 통한 캡션 기능 Timed Text XML 파일을 사용하는 대신 FLV 파일에 포함된 이벤트 큐 포인트를 XML과 연결하여 캡션 기능을 제공합니다.

여러 FLVPlayback 캡션 기능 여러 개의 FLVPlayback 인스턴스에 대해 여러 개의 FLVPlayback 캡션 인스턴스를 만들 수 있습니다.

전환 버튼 컨트롤 캡션 전환 버튼을 통해 캡션 기능과 사용자의 상호 작용을 제공합니다.

FLVPlaybackCaptioning 구성 요소 사용

FLVPlaybackCaptioning 구성 요소는 하나 이상의 FLVPlayback 구성 요소와 사용됩니다. 가장 간단한 사용 방법은 같은 스테이지의 FLVPlayback 구성 요소와 FLVPlaybackCaptioning 구성 요소를 드래그한 후 캡션 URL을 확인하고 표시할 캡션을 설정하는 것입니다. 필요한 경우 다양한 매개 변수를 설정하여 FLVPlayback 캡션을 사용자 정의할 수도 있습니다.

FLVPlayback 구성 요소에 캡션 추가

FLVPlaybackCaptioning 구성 요소는 모든 FLVPlayback 구성 요소에 추가할 수 있습니다. 응용 프로그램에 FLVPlayback 구성 요소를 추가하는 방법에 대한 자세한 내용은 123페이지의 “FLVPlayback 구성 요소를 사용하여 응용 프로그램 만들기”를 참조하십시오.

구성 요소 패널에서 FLVPlaybackCaptioning 구성 요소 추가:

1 [구성 요소] 패널에서 Video 폴더를 엽니다.

2 FLVPlaybackCaptioning 구성 요소를 드래그하거나 두 번 클릭하여, 캡션 기능을 추가할 FLVPlayback 구성 요소와 같은 스테이지에 추가합니다.

참고: Adobe에서 제공하는 두 가지 파일인 caption_video.flv(FLVPlayback 샘플)와 caption_video.xml(캡션 샘플)을 사용하면 FLVPlaybackCaptioning 구성 요소 사용 방법을 쉽고 빠르게 익힐 수 있습니다. 이러한 파일은 www.helpexamples.com/flash/video/caption_video.flv 및 www.helpexamples.com/flash/video/caption_video.xml에서 볼 수 있습니다.

3 (선택 사항) FLVPlayback 및 FLVPlaybackCaptioning 구성 요소와 같은 스테이지에 CaptionButton 구성 요소를 드래그합니다. CaptionButton 구성 요소는 사용자가 캡션을 켜거나 끄는 데 사용됩니다.

참고: CaptionButton 구성 요소를 사용하려면 FLVPlayback 및 FLVPlaybackCaptioning 구성 요소가 있는 스테이지에 이 구성 요소를 드래그해야 합니다.

4 스테이지에서 FLVPlaybackCaptioning 구성 요소를 선택하고 속성 관리자의 [매개 변수] 탭에서 다음 필수 정보를 지정합니다.

- showCaptions를 true로 설정합니다.
- 다운로드할 Timed Text XML 파일의 source를 지정합니다.

Flash에서 캡션을 테스트하는 동안에는 showCaptions 속성을 true로 설정해야 합니다. 그러나 사용자가 캡션을 켜고 끌 수 있도록 CaptionButton 구성 요소를 추가하는 경우에는 showCaptions 속성을 false로 설정해야 합니다.

다른 매개 변수를 사용하여 FLVPlaybackCaptioning 구성 요소를 원하는 대로 사용자 정의할 수 있습니다. 자세한 내용은 164페이지의 “FLVPlaybackCaptioning 구성 요소 사용자 정의” 및 **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**를 참조하십시오.

5 [컨트롤] > [동영상 테스트]를 선택하여 비디오를 시작합니다.

ActionScript를 사용하여 동적으로 인스턴스 만들기:

1 [구성 요소] 패널의 FLVPlayback 구성 요소를 [라이브러리] 패널([윈도우] >[라이브러리])로 드래그합니다.

2 [구성 요소] 패널의 FLVPlaybackCaptioning 구성 요소를 [라이브러리] 패널로 드래그합니다.

3 타임라인의 프레임 1에서 다음 코드를 [액션] 패널에 추가합니다.

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "install_drive:/Program Files/Adobe/Adobe Flash CS5/en/Configuration/FLVPlayback
Skins/ActionScript 3.0/SkinUnderPlaySeekCaption.swf";
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/caption_video.flv";
var my_FLVPlybkcap = new FLVPlaybackCaptioning();
addChild (my_FLVPlybkcap);
my_FLVPlybkcap.source = "http://www.helpexamples.com/flash/video/caption_video.xml";
my_FLVPlybkcap.showCaptions = true;
```

4 **install_drive**를 Flash를 설치한 드라이브로 변경하고 사용자 설치 환경의 Skins 폴더 위치를 가리키도록 경로를 수정합니다.

참고: ActionScript를 사용하여 FLVPlayback 인스턴스를 만들 경우 ActionScript로 스킨 속성을 설정하여 인스턴스에 스킨을 동적으로 할당해야 합니다. ActionScript를 사용하여 스킨을 적용할 경우에는 SWF 파일과 함께 스킨이 자동으로 제작되지 않습니다. 따라서 스킨 SWF 파일과 응용 프로그램 SWF 파일을 서버에 복사하지 않으면 실행 시 스킨 SWF 파일을 사용할 수 없습니다.

FLVPlaybackCaptioning 구성 요소 매개 변수 설정

속성 관리자나 [구성 요소 관리자]에서 FLVPlaybackCaptioning 구성 요소의 각 인스턴스에 대해 다음과 같은 매개 변수를 설정하여 구성 요소를 사용자 정의할 수 있습니다. 다음은 속성 목록과 각각에 대한 간단한 설명입니다.

autoLayout FLVPlaybackCaptioning 구성 요소가 캡션 영역의 크기를 제어하는지 여부를 결정합니다. 기본값은 true입니다.

captionTargetName 캡션이 포함된 TextField 또는 MovieClip 인스턴스의 이름을 나타냅니다. 기본값은 auto입니다.

flvPlaybackName 캡션을 사용할 FLVPlayback 인스턴스의 이름을 나타냅니다. 기본값은 auto입니다.

simpleFormatting true로 설정할 경우 Timed Text XML 파일의 서식 명령을 제한합니다. 기본값은 false입니다.

showCaptions 캡션을 표시할지 여부를 결정합니다. 기본값은 true입니다.

source Timed Text XML 파일의 위치를 나타냅니다.

모든 FLVPlaybackCaptioning 매개 변수에 대한 자세한 내용은 **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**를 참조하십시오.

source 매개 변수 지정

source 매개 변수는 동영상의 캡션이 들어 있는 Timed Text XML 파일의 이름과 위치를 지정하는 데 사용됩니다. [구성 요소 관리자]에서 source 셀에 URL 경로를 직접 입력합니다.

캡션 표시

캡션을 보려면 showCaptions 매개 변수를 true로 설정합니다.

모든 FLVPlaybackCaptioning 구성 요소 매개 변수에 대한 자세한 내용은 **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**를 참조하십시오.

이전 예제에서는 FLVPlaybackCaptioning 구성 요소를 만들고 설정하여 캡션을 표시하는 방법을 배웠습니다. 캡션 소스로 사용할 수 있는 파일에는 두 가지가 있습니다. 그 중 하나는 캡션이 들어 있는 Timed Text XML 파일이며 다른 하나는 포함된 이벤트 큐 포인트에 연결할 수 있는 XML 파일로, 캡션 텍스트가 들어 있습니다.

Timed Text 캡션 사용

FLVPlaybackCaptioning 구성 요소를 사용하면 TT(Timed Text) XML 파일을 다운로드하여 관련 FLVPlayback 구성 요소에 캡션을 사용할 수 있습니다. Timed Text 형식에 대한 자세한 내용은 www.w3.org.의 AudioVideo Timed Text 정보를 참조하십시오.

이 단원에서는 지원되는 Timed Text 태그, 필수 캡션 파일 태그 및 Timed Text XML 파일의 예제를 제공합니다. 지원되는 모든 Timed Text 태그에 대한 자세한 내용은 158페이지의 “Timed text 태그”를 참조하십시오.

FLVPlaybackCaptioning 구성 요소는 다음과 같은 Timed Text 태그를 지원합니다.

범주	작업
단락 서식 지원	단락을 오른쪽, 왼쪽 또는 가운데에 정렬할 수 있습니다.
텍스트 서식 지원	• 절대 픽셀 크기 또는 델타 스타일(예: +2, -4)을 사용하여 텍스트 크기를 설정할 수 있습니다. • 텍스트 색상 및 글꼴을 설정할 수 있습니다. • 텍스트를 굵게 또는 기울임체로 설정할 수 있습니다. • 텍스트 맞춤을 설정할 수 있습니다.
기타 서식 기능 지원	• 캡션을 표시할 TextField의 배경색을 설정할 수 있습니다. • 캡션을 표시할 TextField의 배경색을 투명하게 설정할 수 있습니다(알파 0). • 캡션을 표시할 TextField에 줄 바꿈(on 또는 off)을 설정할 수 있습니다.

FLVPlaybackCaptioning 구성 요소는 FLV 파일의 시간 코드와 일치합니다. 모든 캡션에는 캡션이 표시되는 시점을 결정하는 begin 특성이 있어야 합니다. 캡션에 dur 또는 end 특성이 없으면 캡션은 다음 캡션이 나타날 때 또는 FLV 파일이 끝날 때 사라집니다.

다음은 Timed Text XML 파일의 예제입니다. 이 파일(caption_video.xml)은 caption_video.flv 파일에 캡션을 제공합니다. 이러한 파일은 www.helpexamples.com/flash/video/caption_video.flv 및 www.helpexamples.com/flash/video/caption_video.xml에서 볼 수 있습니다.

```
<?xml version="1.0" encoding="UTF-8"?>
    <tt xml:lang="en"
xmlns="http://www.w3.org/2006/04/ttaf1"xmlns:tts="http://www.w3.org/2006/04/ttaf1#styling">
<head>
    <styling>
<style id="1" tts:textAlign="right"/>
<style id="2" tts:color="transparent"/>
<style id="3" style="2" tts:backgroundColor="white"/>
<style id="4" style="2 3" tts:fontSize="20"/>
    </styling>
</head>
<body>
    <div xml:lang="en">
<p begin="00:00:00.00" dur="00:00:03.07">I had just joined <span
tts:fontFamily="monospaceSansSerif,proportionalSerif,TheOther"tts:fontSize="+2">Macromedia</span> in
1996,</p>
<p begin="00:00:03.07" dur="00:00:03.35">and we were trying to figure out what to do about the internet.</p>
<p begin="00:00:06.42" dur="00:00:03.15">And the company was in dire straights at the time.</p>
<p begin="00:00:09.57" dur="00:00:01.45">We were a CD-ROM authoring company,</p>
<p begin="00:00:11.42" dur="00:00:02.00">and the CD-ROM business was going away.</p>
<p begin="00:00:13.57" dur="00:00:02.50">One of the technologies I remember seeing was Flash.</p>
<p begin="00:00:16.47" dur="00:00:02.00">At the time, it was called <span tts:fontWeight="bold"
tts:color="#ccc333">FutureSplash</span>.</p>
<p begin="00:00:18.50" dur="00:00:01.20">So this is where Flash got its start.</p>
<p begin="00:00:20.10" dur="00:00:03.00">This is smart sketch running on the <span
tts:fontStyle="italic">EU-pin computer</span>,</p>
<p begin="00:00:23.52" dur="00:00:02.00">which was the first product that FutureWave did.</p>
<p begin="00:00:25.52" dur="00:00:02.00">So our vision for this product was to</p>
<p begin="00:00:27.52" dur="00:00:01.10">make drawing on the computer</p>
<p begin="00:00:29.02" dur="00:00:01.30" style="1">as <span tts:color="#ccc333">easy</span> as drawing on
paper.</p>
</div>
</body>
</tt>
```

Timed text 태그

FLVPlaybackCaptioning 구성 요소는 XML 파일에 캡션을 추가하는 데 사용할 수 있는 Timed Text 태그를 지원합니다. 오디오 비디오 Timed Text 태그에 대한 자세한 내용은 www.w3.org의 정보를 참조하십시오. 다음 표에는 지원되는 태그와 지원되지 않는 태그의 목록이 나와 있습니다.

함수	태그/값	용도/설명	예제
무시되는 태그	metadata	무시됨 / 문서의 모든 수준에서 사용할 수 있습니다.	
	set	무시됨 / 문서의 모든 수준에서 사용할 수 있습니다.	
	xml:lang	무시됨	
	xml:space	무시됨 / 비헤이비어가: xml:space="default"로 재정의됩니다.	
	layout	무시됨 / layout 태그 섹션의 모든 region 태그도 함께 무시됩니다.	
	br 태그	모든 특성과 내용이 무시됩니다.	

함수	태그/값	용도/설명	예제
캡션의 미디어 시간	begin 특성	p 태그에만 사용할 수 있습니다. 미디어 시간에 따라 캡션을 배포하는 데 필요합니다.	<p begin="3s">
	dur 특성	p 태그에만 사용할 수 있으며 권장 사항입니다. 이 특성을 포함하지 않으면 FLV 파일이 끝나거나 다른 캡션이 시작할 때 캡션이 끝납니다.	
	end 특성	p 태그에만 사용할 수 있으며 권장 사항입니다. 이 특성을 포함하지 않으면 FLV 파일이 끝나거나 다른 캡션이 시작할 때 캡션이 끝납니다.	
캡션의 시계 시간	00:03:00.1	전체 시간 형식	
	03:00.1	부분 시간 형식	
	10	단위 없는 오프셋 시간. 오프셋은 초를 나타냅니다.	
	00:03:00:05 00:03:00:05.1 30f 30t	지원되지 않습니다. 프레임이나 구분 부호가 있는 시간 형식은 지원되지 않습니다.	
Body 태그	body	필수 태그 / body 태그는 하나만 지원됩니다.	<body> ... </body>
내용 태그	div 태그	사용하지 않거나 원하는 만큼 사용할 수 있습니다. 첫 번째 태그가 사용됩니다.	
	p 태그	사용하지 않거나 원하는 만큼 사용할 수 있습니다.	
	span 태그	일련의 텍스트 내용 단위의 논리 컨테이너입니다. 중첩 span 태그는 지원되지 않습니다. 특성 스타일 태그를 지원합니다.	
	br 태그	명시적인 줄 바꿈을 나타냅니다.	
스타일 태그 모든 style 태그는 p 태그 안에 사용됩니다.	style	하나 이상의 스타일 요소를 참조합니다. 태그 및 특성으로 사용할 수 있습니다. 태그로 사용할 경우에는 ID 특성이 필요하며 해당 스타일을 문서에서 다시 사용할 수 있습니다. style 태그 안에 style 태그를 하나 이상 포함할 수 있습니다.	
	tts:background Color	영역의 배경색을 정의하는 스타일 속성을 지정합니다. 배경을 투명하게 만드는 알파 0을 제외하고는 모든 알파가 무시됩니다. 색상 형식은 #RRGGBBAA입니다.	

함수	태그/값	용도/설명	예제
	tts:color	전경색을 정의하는 스타일 속성을 지정합니다. 모든 색상에 대해 알파가 지원되지 않습니다. transparent 값은 검정으로 해석됩니다.	<style id="3" style="2" tts:backgroundColor="white"/> "transparent" = #00000000 "black"=#000000FF "silver"=#C0C0C0FF "grey"=#808080FF "white"=#FFFFFFFF "maroon"=#800000FF "red"=#FF0000FF "purple"=#800080FF "fuchsia"("magenta")= #FF00FFFF "green"=#008000FF "lime"=#00FF00FF "olive"=#808000FF "yellow"=#FFFF00FF "navy"=#000080FF "blue"=#0000FFFF "teal"=#008080FF "aqua"("cyan")=#00FFFFFF
	tts:fontFamily	글꼴 집합을 정의하는 스타일 속성을 지정합니다.	"default" = _serif "monospace" = _typewriter "sansSerif" = _sans "serif" = _serif "monospaceSansSerif" = _typewriter "monospaceSerif" = _typewriter "proportionalSansSerif" = _sans
	tts:fontSize	글꼴 크기를 정의하는 스타일 속성을 지정합니다. 값을 두 개 제공하면 첫 번째(세로) 값만 사용됩니다. 백분율 값과 단위는 무시됩니다. 절대 픽셀(예: 12) 및 상대 스타일(예: +2) 크기 모두 지원됩니다.	
	tts:fontStyle	글꼴 스타일을 정의하는 스타일 속성을 지정합니다.	"normal" "italic" "inherit"* * 기본 비헤이비어이며 포함하는 태그의 스타일을 상속합니다.

함수	태그/값	용도/설명	예제
	tts:fontWeight	글꼴 두께를 정의하는 스타일 속성을 지정합니다.	"normal" "bold" "inherit"* * 기본 비헤이비어이며 포함하는 태그의 스타일을 상속합니다.
	tts:textAlign	블럭 영역 내의 인라인 영역 정렬 방법을 정의하는 스타일 속성을 지정합니다.	"left" "right" "center" "start" (="left") "end" (="right") "inherit"* *포함하는 태그의 스타일을 상속합니다. textAlign 태그를 설정하지 않는 경우 기본값은 "left"입니다.
	tts:wrapOption	영향을 받는 요소의 컨텍스트에서 자동 줄 바꿈을 적용할지 여부를 정의하는 스타일 속성을 지정합니다. 이 설정은 캡션 요소의 모든 단락에 적용됩니다.	"wrap" "noWrap" "inherit"* *포함하는 태그의 스타일을 상속합니다. wrapOption 태그를 설정하지 않는 경우 기본값은 "wrap"입니다.
지원되지 않는 특성	tts:direction tts:display tts:displayAlign tts:dynamicFlow tts:extent tts:lineHeight tts:opacity tts:origin tts:overflow tts:padding tts:showBackground tts:textOutline tts:unicodeBidi tts:visibility tts:writingMode tts:zIndex		

캡션에 큐 포인트 사용

큐 포인트를 사용하면 FLV 파일 재생에 영향을 주거나 비디오 내의 특정 지점에 텍스트를 표시하는 등 비디오와 상호 작용할 수 있습니다. FLV 파일에 사용할 Timed Text XML 파일이 없을 때는 FLV 파일에 이벤트 큐 포인트를 포함한 다음 해당 큐 포인트를 텍스트에 연결할 수 있습니다. 이 단원에서는 FLVPlaybackCaptioning 구성 요소 큐 포인트 표준에 대한 정보와 이러한 큐 포인트를 텍스트에 연결하여 캡션을 사용하는 방법에 대한 개요를 제공합니다. 비디오 가져오기 마법사나 Flash Video Encoder를 사용하여 이벤트 큐 포인트를 포함하는 방법에 대한 자세한 내용은 **Flash 사용**의 16장, "비디오를 사용한 작업"을 참조하십시오.

FLVPlaybackCaptioning 큐 포인트 표준

FLV 파일의 메타데이터에서 큐 포인트는 name, time, type 및 parameters 속성이 있는 객체로 표현됩니다. FLVPlaybackCaptioning ActionScript 큐 포인트의 특성은 다음과 같습니다.

name name 속성은 큐 포인트의 이름을 포함하는 문자열입니다. name 속성은 **fl.video.caption.2.0.** 접두어로 시작하고 그 다음에 문자열이 와야 합니다. 문자열은 각 이름을 고유하게 식별하기 위해 하나씩 증가하는 일련의 정수입니다. 접두어에는 FLVPlayback 버전 번호와 일치하는 버전 번호도 포함됩니다. Adobe Flash CS4 이상의 경우 버전 번호를 2.0으로 설정해야 합니다.

time time 속성은 캡션이 나타나야 하는 시간입니다.

type type 속성은 값이 "event"인 문자열입니다.

parameters parameters 속성은 다음과 같은 이름-값 쌍을 지원하는 배열입니다.

- text:String 캡션으로 사용할 HTML 형식의 텍스트입니다. 이 텍스트는 TextField.htmlText 속성에 직접 전달됩니다. FLVPlaybackCaptioning 구성 요소는 다국어 트랙을 사용할 수 있도록 하는 선택적인 text:**n** 속성을 지원합니다. 자세한 내용은 163페이지의 “포함된 큐 포인트를 통해 다국어 트랙 지원”을 참조하십시오.
- endTime:Number 캡션이 사라지는 시간입니다. 이 속성을 지정하지 않으면 FLVPlaybackCaptioning 구성 요소는 해당 속성이 숫자가 아닌 것(NaN)으로 간주하고 캡션은 FLV 파일이 완료될 때까지 표시됩니다(FLVPlayback 인스턴스가 VideoEvent.COMPLETE 이벤트를 전달). endTime:Number 속성은 초 단위로 지정합니다.
- backgroundColor:uint 이 매개 변수는 TextField.backgroundColor를 설정합니다. 이 속성은 선택 사항입니다.
- backgroundColorAlpha:Boolean backgroundColor의 alpha가 0%이면 이 매개 변수는 TextField.background를 !backgroundColor로 설정합니다. 이 속성은 선택 사항입니다.
- wrapOption:Boolean 이 매개 변수는 TextField.wordWrap을 설정하며 선택 사항입니다.

포함된 이벤트 큐 포인트에 캡션 사용

FLV 파일에 대한 캡션이 들어 있는 Timed Text XML 파일이 없는 경우에는 캡션이 들어 있는 XML 파일을 포함된 이벤트 큐 포인트에 연결하여 캡션을 만들 수 있습니다. XML 샘플은 다음과 같은 단계를 수행하여 비디오에 포함된 이벤트 큐 포인트를 이미 만들었다고 가정합니다.

- FLVPlaybackCaptioning 표준에 따라 이벤트 큐 포인트를 추가하고 비디오를 인코딩합니다.
- Flash에서 FLVPlayback 구성 요소와 FLVPlaybackCaptioning 구성 요소를 스테이지로 드래그합니다.
- FLVPlayback 및 FLVPlaybackCaptioning 구성 요소의 source 속성(FLV 파일 및 XML 파일 위치)을 설정합니다.
- 제작합니다.

  다음은 XML을 인코더로 가져오는 샘플입니다.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<FLVCoreCuePoints>

    <CuePoint>
        <Time>9136</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index1</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the first cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

    <CuePoint>
        <Time>19327</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index2</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the second cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

    <CuePoint>
        <Time>24247</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index3</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the third cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

    <CuePoint>
        <Time>36546</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index4</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the fourth cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

</FLVCoreCuePoints>
```

FLVPlaybackCaptioning 구성 요소는 포함된 큐 포인트를 통해 다국어 트랙도 지원합니다. 자세한 내용은 163페이지의 “포함된 큐 포인트를 통해 다국어 트랙 지원”을 참조하십시오.

포함된 큐 포인트를 통해 다국어 트랙 지원

FLVPlaybackCaptioning track 속성은 Timed Text XML 파일이 FLVPlaybackCaptioning 큐 포인트 표준을 따를 경우 포함된 큐 포인트를 사용하여 다국어 트랙을 지원합니다. 자세한 내용은 162페이지의 “FLVPlaybackCaptioning 큐 포인트 표준”을 참조하십시오. 그러나 별도의 XML 파일을 사용할 경우에는 FLVPlaybackCaptioning 구성 요소가 다국어 트랙을 지원하

지 않습니다. track 속성을 사용하려면 이 속성을 0이 아닌 값으로 설정해야 합니다. 예를 들어, track 속성을 1(track == 1)로 설정하면 FLVPlaybackCaptioning 구성 요소는 큐 포인트 매개 변수를 검색합니다. 이때 일치하는 항목이 없으면 큐 포인트 매개 변수의 text 속성이 사용됩니다. 자세한 내용은 **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**의 track 속성을 참조하십시오.

캡션이 있는 여러 FLV 파일 재생

하나의 FLVPlayback 구성 요소 인스턴스 내에서 비디오 플레이어를 여러 개 열어 다수의 비디오를 재생하고 재생 중인 비디오 사이를 전환할 수 있습니다. 또한 FLVPlayback 구성 요소 내의 각 비디오 플레이어에 캡션을 연결할 수 있습니다. 여러 비디오 플레이어를 여는 방법에 대한 자세한 내용은 135페이지의 "여러 비디오 플레이어 사용"을 참조하십시오. 여러 비디오 플레이어에 캡션을 사용하려면 각 VideoPlayer에 대해 FLVPlaybackCaptioning 구성 요소 인스턴스를 하나씩 만들고 FLVPlaybackCaptioning videoPlayerIndex를 해당 인덱스로 설정해야 합니다. VideoPlayer가 하나만 있는 경우 VideoPlayer 인덱스는 기본적으로 0으로 지정됩니다.

다음은 고유한 비디오에 고유한 캡션을 할당하는 코드 예제입니다. 이 예제를 실행하려면 예제에 나와 있는 가공의 URL을 실제로 작동하는 URL로 바꾸어야 합니다.

```
captioner0.videoPlayerIndex = 0;
captioner0.source = "http://www.[yourDomain].com/mytimedtext0.xml";
flvPlayback.play("http://www.[yourDomain].com/myvideo0.flv");
captioner1.videoPlayerIndex = 1;
captioner1.source = "http://www.[yourDomain].com/mytimedtext1.xml";
flvPlayback.activeVideoIndex = 1;
flvPlayback.play ("http://www.[yourDomain].com/myvideo1.flv");
```

FLVPlaybackCaptioning 구성 요소 사용자 정의

캡션을 FLVPlayback 구성 요소 위에 배치하는 FLVPlaybackCaptioning 기본값을 사용하면 FLVPlaybackCaptioning 구성 요소를 신속하게 시작할 수 있습니다. 그러나 비디오와 떨어진 곳으로 캡션을 이동하도록 FLVPlaybackCaptioning 구성 요소를 사용자 정의하는 것이 좋습니다.

다음 코드에서는 캡션 전환 버튼을 사용하여 FLVPlayback 객체를 동적으로 만드는 방법을 보여 줍니다.

1. FLVPlayback 구성 요소를 스테이지의 0,0 지점에 배치하고 인스턴스 이름을 **player**로 지정합니다.
2. FLVPlaybackCaptioning 구성 요소를 스테이지의 0,0에 배치하고 인스턴스 이름을 **captioning**라고 지정합니다.
3. 스테이지에 CaptionButton 구성 요소를 배치합니다.
4. 다음의 코드 예제에서는 testVideoPath:String 변수를 FLV 파일로 설정합니다(절대 또는 상대 경로 사용).

   참고: 이 코드 예제에서는 testVideoPath 변수를 Flash 비디오 샘플인 caption_video.flv로 설정합니다. 이 변수를 캡션 버튼 구성 요소를 추가하려는 캡션 기능 비디오 구성 요소의 경로로 변경하십시오.

5. 다음 코드 예제에서는 testCaptioningPath:String 변수를 적절한 Timed Text XML 파일로 설정합니다(절대 또는 상대 경로 사용).

   참고: 이 코드 예제에서는 testCaptioningPath 변수를 Timed Text XML 파일인 caption_video.xml로 설정합니다. 이 변수를 비디오 캡션이 들어 있는 Timed Text XML 파일의 경로로 변경하십시오.

6. 다음 코드를 FLA 파일과 같은 디렉토리에 FLVPlaybackCaptioningExample.as로 저장합니다.
7. FLA 파일의 DocumentClass를 FLVPlaybackCaptioningExample로 설정합니다.

```
package
{
    import flash.display.Sprite;
    import flash.text.TextField;
    import fl.video.FLVPlayback;
    import fl.video.FLVPlaybackCaptioning;

    public class FLVPlaybackCaptioningExample extends Sprite {

        private var testVideoPath:String = "http://www.helpexamples.com/flash/video/caption_video.flv";
        private var testCaptioningPath:String =
"http://www.helpexamples.com/flash/video/caption_video.xml";

        public function FLVPlaybackCaptioningExample() {
            player.source = testVideoPath;
            player.skin = "SkinOverAllNoCaption.swf";
            player.skinBackgroundColor = 0x666666;
            player.skinBackgroundAlpha = 0.5;

            captioning.flvPlayback = player;
            captioning.source = testCaptioningPath;
            captioning.autoLayout = false;
            captioning.addEventListener("captionChange",onCaptionChange);
        }
        private function onCaptionChange(e:*):void {
            var tf:* = e.target.captionTarget;
            var player:FLVPlayback = e.target.flvPlayback;

            // move the caption below the video
            tf.y = 210;
        }
    }
}
```

모든 FLVPlaybackCaptioning 매개 변수에 대한 자세한 내용은 **Adobe® Flash® Professional CS5용 ActionScript® 3.0 참조 설명서**를 참조하십시오.